TECHNICAL REPORT NO. 11

"MARKOVIAN DECISION PROCESSES WITH UNCERTAIN TRANSITION PROBABILITIES"

by

2 John M. Cozzolino
Romulo Gonzalez-Zubieta
Ralph L. Miller

March, 1965

RESEARCH IN THE CONTROL OF COMPLEX SYSTEMS

MASSACHUSETTS INSTITUTE
OF
TECHNOLOGY

MARKOVIAN DECISION PROCESSES WITH UNCERTAIN TRANSITION PROBABILITIES

by

John M. Cozzolino

Romulo Gonzalez-Zubieta

Ralph L. Miller

Technical Report No. 11

On Work Performed Under

Contract Nonr-1841 (87), NR 042-230

Office of Naval Research

RESEARCH IN THE CONTROL OF COMPLEX SYSTEMS

Operations Research Center
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
Cambridge 39, Massachusetts

March, 1965

FOREWORD

The Center for Operations Research at the Massachusetts Institute of Technology is an inter-departmental activity devoted to graduate training and research in the field of operations research. Its products are books, journal articles, detailed reports such as this one, and students trained in the theory and practice of operations research.

The work of the Center is supported, in part, by government contracts and industrial grants-in-aid. The research which led to this particular report was supported by the Office of Naval Research under Contract Nonr-1841(87). Reproduction in whole or in part is permitted for any purpose of the United States Government.

Philip M. Morse,
Director of the Center

Ronald A. Howard,
Project Supervisor

ACKNOWLEDGMFNT

We wish tc express our sincere gratitude to Professor Ronald A. Howard for his guidance and encouragement as project supervisor, and to Dr. George R. Murray, Jr. for his help and advice. We would also like to thank Miss Priscilla A. Mullin and Mrs. Nobu McNeill for their excellent work in organizing and typing of this report. The computer simulations were done with the facilities of the M. I. T. Computation Center.

MARKOVIAN DECISION PROCESSES WITH UNCERTAIN TRANSITION PROBABILITIES

by

John Cozzolino
Romulo Gonzalez
Ralph Miller

ABSTRACT

The theory for finding optimal policies for Markov processes with transition rewards and many alternatives in each state, when the transition probabilities are given, has already been developed. But in most practical applications, these transition probabilities are not known exactly--one has only some prior knowledge about them.

This problem of uncertain transition probabilities was first treated by Dr. E. A. Silver in the Interim Technical Report #1 of the Operations Research Center of M. I. T. Section I of the present report extends several of Silver's results. We propose a dynamic programming formulation for the problem of choosing an optimal operating strategy and we carry out the solution for a special two-state example. However, it is found that solution of non-trivial problems of any higher dimension is impractical.

Section II is concerned with experimental and heuristic approaches to the problem, and relies upon simulation rather than upon analysis. We investigate certain statistics of the process when the unknown transition probabilities are governed by a multi-dimensional beta prior (a convenient form for Bayes modification). We find that the process with known probabilities which are equal to the mean values of the unknown probabilities, provides us with a remarkably good picture of the unknown process. The hypothesis is stated and investigated that this process of expected values is adequate for decision purposes, and that determining decisions from it is feasible as well as useful.

Finally, some alternative approac!es are suggested for cases which might not be handled by the expected values technique.

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

1.1 Statement of problem

The process which we are considering is a multistate discrete time Markov process with transition rewards. The decision structure consists of a set of possible alternative actions in each state. Each such action has associated with it a unique set of transition probabilities and rewards. At each time period, an alternative must be specified for the state currently occupied. A set of alternatives, one for each state, is called a policy.

We might illustrate the structure of the problem by setting forth a simple example that will be familiar to those who have read reference (1). A taxi driver operates between three towns: A, B, and C. The probability of his picking up a fare to a particular destination is dependent upon where he is now, so we set up a Markovian model. But in addition we assume that he can follow one of three different courses of action:

1. He may cruise around and wait to be hailed.
2. He may wait at a cab stand for a fare to come along.
3. He may wait at a radio call box for a call to come through.

Of course, his choice of alternative will alter his probabilities of picking up fares to the various towns--for instance, it may be more likely that a radio call will be for a long trip. Also, the rewards will be influenced by the alternatives because of the differing costs involved--it is cheaper to wait at a cab stand than to cruise (although it may take longer to get a fare).

Let us assume that all three alternatives are open to our driver, except in town B where there is no radio call box, so that alternative 3 is not possible. We might find the following probabilities and rewards to prevail:

Probability Matrix

		Town A	Town B	Town C
P = Town A	1	1/2	1/4	1/4
	2	1/16	3/4	3/16
	3	1/4	1/8	5/8
Town B	1	1/2	0	1/2
	2	1/16	7/8	1/16
Town C	1	1/4	1/4	1/2
	2	1/8	3/4	1/8
	3	3/4	1/16	3/16

Reward Matrix

		Town A	Town B	Town C
R = Town A	1	10	4	8
	2	8	2	4
	3	4	6	4
Town B	1	14	0	18
	2	8	16	8
Town C	1	10	2	8
	2	6	4	2
	3	4	0	6

This then is the structure of the problem we shall consider. When all transition probabilities and rewards are known, it is possible to find the optimal policy for an operation which lasts for a finite time with terminal rewards, or one which lasts indefinitely long. Also, in this infinite duration case, we might maximize the expected gain per period, or, alternatively we might prefer to discount the future rewards and maximize the present value of the infinitely long reward stream. Both of these cases have been solved by Howard's policy iteration method. (1)

The work reported in this technical note concerns the Markov decision problem in which the decision maker has impe. fect knowledge of the transition probabilities for the process. We assume that we have some prior knowledge about these unknown transition probabilities, and that this information is expressed in a probabilistic manner. We can gain information about these unknowns by observing transitions as the process continues.

This problem become: triviai in the infinite duration, no discounting case. The optimal solution to the decision problem is to experiment indefinitely long with every alternative to find the exact values of the unknown transition probabilities, then solve the deterministic decision problem by Howard's policy iteration to find a policy to use for an infinite amount of time. This follows because even an infinitesimal improvement in the expected gain per period is worth an infinite amount in this case.

However, when time has value, this becomes quite a different (and difficult) problem. We must do some experimentation to find the best policy, but the longer we experiment, the less that best policy becomes worth to us. We would expect that for any discount factor, there would exist a best strategy in an expected value sense. Such a strategy would specify the best choice of alternative in each state for each distinct set of prior knowledge. Since prior knowledge is updated in time as the process operates (giving us information), the best choice of alternative changes in time as well. It is important to see that the optimal strategy since it is optimal in an expected value sense, would not necessarily ever lead us to follow that policy which is in fact optimal (that is, we would follow if we knew all the transition probabilities exactly). The optimal strategy will, rather, give the best trade-off between search and immediate earnings.

In this report we will consider a variety of approaches to this problem. Heuristic approaches useful for large problems and analytic methods suitable for small problems will be examined.

1.2 Notation

The discrete time Markov process with N states and with only one alternative in each state is specified by a transition probability matrix $\mathbf{P}$, whose element p_{ij} gives the conditional probability of a transition from i to j given that the process is in state i.

The reward structure is specified by a reward matrix $\mathbf{R}$ whose element r_{ij} gives the reward earned by the process for making a transition from state i to state j.

The steady state probabilities for this process are denoted by a vector $\underline{\pi} = (\pi_1, \pi_2, \ldots\ldots \pi_N.)$ (we assume completely ergotic processes.)

The expected reward on the next transition, given that we are in state i, is given by:

$$q_i = \sum_j p_{ij}\, r_{ij}$$

Finally, the steady state gain, which is expected reward per transition, is:

$$g = \sum_i \pi_i\, q_i$$

If discounting is used, so that the present value of one dollar to be received one period in the future is β then the expected present discounted value of the infinite stream generated by the one policy process starting in state i is denoted by v_i. Clearly v_i depends on the starting state since rewards in the near future are the most important. The column vector, $\underline{v}$, of these values v_i is found from the $\mathbf{P}$ matrix and the $\underline{q}$ vector:

$$\underline{v} = [I - \beta P]^{-1} \underline{q}$$

Now, let there be k_i different alternatives in state i, for i = 1, 2,.....N. There are then $k_1 \cdot k_2 \cdots k_N$ possible policies under which the process may operate. For each policy there is a P matrix, a $\underline{\pi}$ vector, and a $\underline{v}$ vector. We denote alternatives with a lower case superscript, so that p_{ij}^k is the conditional probability of a transition to state j given that we are in state i and operating under alternative k. We use upper case superscripts to denote policies, so $\underline{\pi}^A$ is, the $\underline{\pi}$ vector for policy A.

Finally, the entire P matrix for the problem has $\sum_{i=1}^{N} k_i$ rows, and each entry is an unknown about which we have some prior knowledge. Because we will soon need some precise terminology to distinguish just what process we are referring to, we decide to call this process with unknown transition probabilities the <u>primary process</u>, since it is the primary focus of our concern. We call the entries <u>random variables</u>, over which we have some <u>prior distribution</u>. This is the Bayesian formulation of unknowns.

Because the statistics associated with the primary process are random variables, we write them with a tilde above them. Thus, $\tilde{p}_{ij}^k$ is the unknown probability of going to j under alternative k given that we are in state i. These are random variables and each one has some distribution, we shall select such a distribution from a known parametric family. Thus with each random variable is associated one or more prior parameters. These are just numbers, and for reasons which will become clear in the next chapter, we designate the prior parameters m_{ij}^k.

Later it will be convenient to speak of another process, whose transition probabilities are known exactly and are equal to the mean value of the corresponding probabilities for the primary process. We will call this process the *expected process*. Because we shall refer to it so often, statistics associated with this process will be written without extra tildes or superscripts. Thus by definition:

$$p_{ij}^{k} = E\,(\tilde{p}_{ij}^{k})$$

and π_i is the steady state probability of being in state i for the expected process. Note that since $\tilde{\pi}_j$ is not in general linear in $\tilde{p}_{ij}$, it is not necessary that $\pi_j = E\,(\tilde{\pi}_j)$.

Since the primary process is composed of random variables, we will often find it useful (for simulation purposes) to draw sample points from these distributions to get processes which we will call sample processes. Statistics associated with these processes will be denoted by a *presuperscript* *s*. Thus ${}^{s}\pi_i^{A}$ is the steady state probability of being in state i under policy A for *a particular* sample process s .

Thus, variables associated with the primary process have tildes, variables associated with the expected process have no extra markings, and variables associated with sample processes have a presuperscript s. Because understanding of the relationship of these various processes is so important to what follows, we conclude this section with a graphical portrait of their relationship.

We have an actual process, a true state of nature, but unfortunately, we do not know it exactly:

$$\begin{array}{c}\text{Actual process}\\ \begin{bmatrix}\text{unknown}\\ \text{numbers}\end{bmatrix}\end{array}$$

Being Bayesians, we treat the unknown transition probabilities as random variables, $\tilde{p}^{k}_{ij}$, whose distribution is indicated by a parameter m^{k}_{ij}. Thus:

$$\begin{array}{cc}
\text{primary process} & \text{prior parameters} \\
\begin{bmatrix}
\tilde{p}^{1}_{11} & \tilde{p}^{1}_{12} \cdot\cdot & \tilde{p}^{1}_{1N} \\
\tilde{p}^{2}_{11} & \cdots & \\
\vdots & & \\
\hline
\vdots & & \\
\tilde{p}^{1}_{21} & \cdots & \\
\vdots & & \\
\hline
\vdots & & \\
\tilde{p}^{k_N}_{N1} & \cdots &
\end{bmatrix}
&
\begin{bmatrix}
m^{1}_{11} & m^{1}_{12} \cdots\cdots & m^{1}_{1N} \\
m^{2}_{11} & \cdots & \\
\vdots & & \\
\hline
\vdots & & \\
m^{1}_{21} & \cdots & \\
\vdots & & \\
\hline
\vdots & & \\
m^{k_N}_{N1} & \cdots &
\end{bmatrix} \\
\text{(random variables)} & \text{(known numbers)}
\end{array}$$

For convenience in later analysis we define still another process:

$$\begin{array}{cc}
\text{expected process} & \text{expected process} \\
\begin{bmatrix}
E(\tilde{p}^{1}_{11}) \cdots E(\tilde{p}^{1}_{1N}) \\
E(\tilde{p}^{2}_{11}) \cdots \\
\vdots \\
\hline
E(\tilde{p}^{1}_{21}) \cdots \\
\vdots \\
\hline
\vdots
\end{bmatrix}
&
\begin{bmatrix}
p^{1}_{11} \cdots\cdots p^{1}_{1N} \\
p^{2}_{11} \cdots\cdots \\
\vdots \\
\hline
p^{1}_{21} \cdots\cdots \\
\vdots \\
\hline
\vdots
\end{bmatrix} \\
\text{(known numbers)} & \text{(known numbers)}
\end{array}$$

Finally from the prior distribution we can draw:

sample process 1

$$\left[\begin{array}{llll}
{}^{s_1}p^{1}_{11} & {}^{s_1}p^{1}_{12} & \cdots & {}^{s_1}p^{1}_{1N} \\
{}^{s_1}p^{2}_{11} & \cdots & & \\
\vdots & & & \\
\hdashline
\vdots & & & \\
{}^{s_1}p^{1}_{21} & \cdots & & \\
\vdots & & & \\
\hdashline
\vdots & & & \\
{}^{s_1}p^{k_N}_{N1} & \cdots & &
\end{array}\right]$$

(known numbers)

sample process 2

$$\left[\begin{array}{llll}
{}^{s_2}p^{1}_{11} & {}^{s_2}p^{1}_{12} & \cdots & {}^{s_2}p^{1}_{1N} \\
{}^{s_2}p^{2}_{11} & \cdots & & \\
\vdots & & & \\
\hdashline
\vdots & & & \\
{}^{s_2}p^{1}_{21} & \cdots & & \\
\vdots & & & \\
\hdashline
\vdots & & & \\
{}^{s_2}p^{k_N}_{N1} & \cdots & &
\end{array}\right]$$

(known numbers)

sample process 3

$$\left[\begin{array}{llll}
{}^{s_3}p^{1}_{11} & {}^{s_3}p^{1}_{12} & \cdots & {}^{s_3}p^{1}_{1N} \\
{}^{s_3}p^{2}_{11} & \cdots & & \\
\vdots & & & \\
\hdashline
\vdots & & & \\
{}^{s_3}p^{1}_{21} & \cdots & & \\
\vdots & & & \\
\hdashline
\vdots & & & \\
{}^{s_3}p^{k_N}_{N1} & \cdots & &
\end{array}\right]$$

(known numbers)

CHAPTER II

THE MULTI-DIMENSIONAL BETA DISTRIBUTION

2.1 In search of a prior distribution

The problem we have formulated assumes that there is prior knowledge of the unknown transition probabilities of the primary process, and that this knowledge is to be expressed probabilistically. We shall now display a convenient form in which to express this knowledge.

Assume that at some moment the system is in state i. The immediate future of the system can be described as a multi-nomial Bernoulli process, That is, one and only one transition to another state will take place, and that transition will be made according to the probabilities p_{ij}, $j = 1, 2. \dots N$. If we consider only those transitions made out of state i we have a multinomial process:

$$pr(E_i \mid p_{i1}, p_{i2}, \dots\dots\dots, p_{iN}) = C \cdot p_{i1}^{n_1} \cdot p_{i2}^{n_2} \dots\dots p_{iN}^{n_N}$$

Where E_i is the event of observing exactly n_1 transitions from state i to state 1, n_2 from i to 2, etc. The conjugate prior for this distribution is the multidimensional beta distribution:

$$f_\beta(p_{i1} \dots\dots p_{iN} \mid m_{i1} \dots\dots\dots m_{iN}) = C \cdot p_{i1}^{m_{i1}-1} \dots\dots p_{iN}^{m_{iN}-1}$$

We shall shortly show that the multi-dimensional beta is an exceptionally convenient form to use for the distribution for $\tilde{p}_{i1}, \tilde{p}_{i2} \dots\dots\dots$ But first it must be noted that use of this prior involves the implicit assumption that the random variables in each row of the transition matrix are independent of the random variables in any other row of the matrix.

That this would in fact be the case in any application is not obvious, and in some sense it is even unlikely. In the example proposed in the introduction, a similar type of alternative existed in every state. If those probabilities were unknown, information about alternative 1 in state 1 might well give us some clues as to the transition probabilities for the same alternative in state 2. So, while we shall use the multidimensional beta distribution throughout, in order to keep the calculations simple, it should be borne in mind that this implicit assumption of independence may not be desirable in some applications.

Throughout this report we shall denote by m_{ij} the beta parameters used, and shall denote by N_i the row sum of the m_{ij}:

$$N_i = \sum_j m_{ij} \qquad\qquad i = 1, 2, \ldots\ldots, N$$

We shall now consider some of the properties of this distribution, and an interpretation for the m_{ij}.

2.2 Properties of the multidimensional beta distribution

We simply list some of the general properties of the multidimensional beta distribution.

1. The marginal distribution of a particular $\tilde{p}_{ij}$ is given by:

   $$f_\beta (p_{ij} \mid m_{ij}, N_i - m_{ij})$$

2. The expected value of a particular $\tilde{p}_{ij}$ is given by:

   $$\bar{p}_{ij} = \mathbf{E} (\tilde{p}_{ij}) = \frac{m_{ij}}{N_i}$$

3. The variance of a particular $\tilde{p}_{ij}$ is given by:

   $$\overset{\vee}{p}_{ij} = \frac{m_{ij}}{N_i} \left(1 - \frac{m_{ij}}{N_i} \right) \frac{1}{N_i + 1}$$

   $$= \bar{p}_{ij} (1 - \bar{p}_{ij}) (N_i + 1)^{-1}$$

4. Bayes modification of the distribution can be accomplished by inspection. Let E be the event that we observe f_{ij} transitions from i to j, $j = 1, 2, \ldots\ldots N$. Then the posterior distribution is:

$$f_{\beta}\,(p_{i1},\ p_{i2}, \ldots\ldots p_{iN} \mid m_{i1} + f_{i1},\ m_{i2} + f_{i2} \ldots\ldots m_{iN} + f_{iN})$$

2.3 Determination of prior parameters

The expression for the mean $\bar{p}_{ij}$ gives us an interpretation of the prior parameters, and thus an intuitive way of assigning them. Since $E\,(\tilde{p}_{ij}) = \dfrac{m_{ij}}{N_i}$, we can specify, instead of the m_{ij}, N-1 of the mean values and a value for N_i. From these statistics, the prior parameters can be calculated. N_i is some kind of a measure of our certainty, as we can see by noting that the variance is inversely proportional to $N_i + 1$. E. A. Silver (3) notes that good results can be obtained by a least squares fit of the intuitive marginal variances to a single value N_i. But in most applications it is likely that one will not have a good idea of the variances which are to be so fit -- so N_i will remain a somewhat crude measure of "certainty."

It may help to note that specifying an initial value of N_i will yield the same result as if we had specfied a uniform prior, and had taken then N_i - N observations which happened to yield the same mean values. So that specifying a value of N_i, is in a sense equivalent to having taken N_i - N observations starting from a uniform prior.

2.4 Methods of sampling

Simulation methods will be used extensively in the following work; it is therefore important to have a method of sampling from the beta prior. Ideally, the prior represents completely our knowledge about the state of nature, so that the actual process is only a "random draw from the prior."

We take this interpretation literally, and test all our heuristics by drawing many sample processes from the prior and evaluating the heuristics with each one of the possible states of nature thus obtained. The sampling procedure is an approximation to the usual Bayesian technique of pre-posterior analysis which, in the examples we shall study, is an exceedingly complex mathematical task.

E. A. Silver (3) has shown that to randomly draw from $f_\beta(\underline{p} \mid \underline{m})$, we can take independent draws (y_{ij}) from the N simple Gamma distributions:

$$f_\gamma(y_{ij} \mid m_{ij}, q) = \frac{q^{m_{ij}}}{\Gamma(m_{ij})} \, y_{ij}^{m_{ij}-1} \, e^{-q y_{ij}} \qquad 0 \leq y_{ij} \leq \infty$$

and then $${}^{s}p_{ij} = \frac{y_{ij}}{\sum_{j} y_{ij}}$$

If all the m_{ij} are integral, the Gamma becomes a simple Erlang-m_{ij}.

Mosiman (2) shows that to sample from the Erlang-m_{ij}:

$$y_{ij} = \frac{1}{q} \sum_{i=1}^{m_{ij}} | \ln r_i | = - \frac{1}{q} \sum_{i=1}^{m_{ij}} \ln r_i ,$$

where r_i is drawn from a rectangular distribution on [0, 1]. Since we divide by $\sum_{j} y_{ij}$, the scale factor q is irrelevant, and can be chosen to keep the logarithms in a convenient range (we use q = 20.) To be certain that all the m_{ij} are integral, we truncate the actual values. This is necessary, as Mosiman also states that there is no way to sample from a general Gamma, short of using tables of the incomplete Gamma function.

The subroutine GEN in Appendix A shows a programmed version of this sampling routine.

CHAPTER III

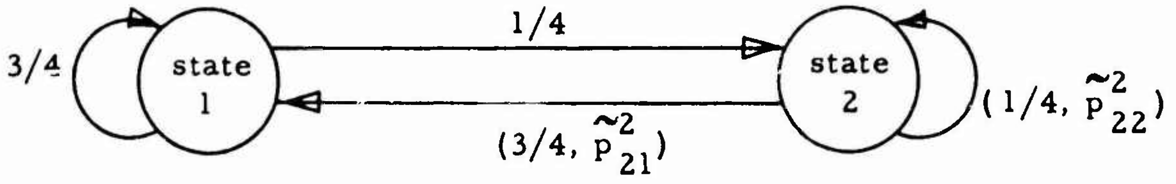
A SPECIAL TWO STATE PROBLEM

3.1 The Problem

We now have a notation for expressing our decision problem, and a prior distribution to encode our knowledge about the unknown transition probabilities. Let us now begin to formulate the general decision problem described in the introduction.

We initiate this task by considering a very simple special case, preparatory to the formulation of the most general case in Chapter IV.

Consider the following two-state discrete time Markov process. When in state 1 there is no choice; the transition probabilities are $(P_{11}, P_{12}) = (3/4, 1/4)$, and the transition rewards (r_{11}, r_{12}). There are two alternatives when in state 2; alternative 1 having transition probabilities $(P^1_{21}, P^2_{22}) = (1/4, 3/4)$ with rewards (r^1_{21}, r^2_{22}), and alternative 2 having the unknown transition probabilities $(\tilde{p}^2_{21}, \tilde{p}^2_{22})$ with known rewards (r^2_{21}, r^2_{22}). This process can be represented by the following diagram:



We assume that the random variables $\tilde{p}_{21}^{2}$ and $\tilde{p}_{22}^{2}$ are multi-dimensional beta distributed. For simplicity, we will denote $\tilde{p}_{21}^{2}$ simply by p for the remainder of this chapter, as there are no other unknown probabilities. ($\tilde{p}_{22}^{2} = 1 - p$). We denote the beta parameter m_{21}^{2} simply by r, and the parameter N_{2}^{2} simply by n. ($m_{22}^{2} = N_{2}^{2} - m_{21}^{2}$)

The decision problem now becomes to specify for each set of parameters (r, n) whether to follow alternative 1 or alternative 2 in state 2.

3.2 Formulation by Dynamic Programming

3.2.1 The Dynamic Programming Approach

The solution of a problem of this type depends upon all possible outcomes and all decisions in the future. The outcome and decision tree, however, is infinite since the process will operate for an indefinitely long time. Discounting reduces the importance of the future, but we are essentially dealing with a boundary value problem with the boundary of infinity. The only technique presently known for solving such a problem is that of truncation to a finite outcome decision space. This finite problem can be solved by dynamic programming, and the size of the finite case taken large enough to yield a converging approximation to the infinite case. Usually this must be done numerically, though it may be possible to discover certain analytic properties of the optimum solution which enable us to find the optimum by another, possibly analytic, method.

3.2.2. Terminal Policy

In order to formulate the problem we will first introduce a terminal situation. Suppose we are given a process which will operate for an indefinitely long period of time. While the process operates we must make a decision every time that the process enters any state with more than one alternative. We will observe the outcomes and improve our decisions in time. However, suppose that at some preknown time we will make a terminal policy decision which will be used throughout the remainder of the operation. This terminal decision policy will specify not only an alternative for the state currently occupied, but also alternative decisions for all states which might be entered in the future. The alternative to be chosen at each state is always fixed (i.e. the terminal policy is time-invariant.) Each possible terminal policy decision has a terminal value associated with it.

3.2.3. The Decision Stage

The time from the start of the process until the terminal decision will consist of a fixed number of decision stages. One possible choice for the decision stage would be the period (holding time) of the process. But, as decisions are only made when in state 2 for this simple process, it will be convenient to define the decision stage to be the time between consecutive departures from state 2. This time is random in terms of the number of transitions, but this disadvantage is more than offset by the fact that with this choice we know precisely how many observations of the transition whose probability is unknown are taken per decision stage--precisely one. Anytime the process enters state 1 it will be regarded as having the decision stage number of the next entry into state 2.

3.2.4 Dynamic Programming Equations

At the 0th stage we will choose the policy with the highest expected terminal value. Let $V_2^N(r,n)$ denote that expected present discounted value of the infinitely long income stream from the process given that an optimal policy is followed, the system is in state 2 and N more decisions are to be made before the terminal decision. This quantity is obviously a function of the prior parameters of the unknown p. We use a capital V for this value function which has prior parameters as arguments to distinguish it from the function lower case v which is the value function when the probabilities are known. We can always compute v from the equations (see Chapter I). $\underline{v} = [I - \beta \mathbf{P}]^{-1} \underline{q}$. Capital V is another matter. At the 0th stage, we can then write:

$$V_2^0 \; (r, n) = \text{Max} \; \{ v_2 (P^1), \int_{P^2} v_2 (P^2) f_\beta (P^2 \mid r, n) \, d P^2 \}$$

where P^k denotes the square transition probability matrix associated with policy k (i.e. follow alternative k when in state 2.) The first term inside the bracket is the value, when in state 2, of following policy 1. The second term is the expected value, when in state 2, of following policy 2.

We can next write expressions for V_2^N (r, n) in terms of similar functions for (N - 1). Given that we are in state 2 with N stages left, with probability p we will be in state 1 with (N - 1) stages left, and with probability 1-p we will be in state 2 with (N-1) stages to go. Taking the expectation over p (recall that $\bar{p} = r/n$), we find:

$$V_2^N (r, n) = \text{Max} \; \{ \; v_2 (P^1), \; \frac{r}{n} \; [\, r_{21}^2 + \beta \, V_1^{N-1} (r + 1, \, n + 1)]$$
$$+ (1 - \frac{r}{n}) \, [\, r_{22}^2 + \beta \, V_2^{N-1} (r, n+1)] \; \}$$

Notice that the first expression inside the braces, which corresponds to taking alternative 1, is a very simple expression. This is because once we decide to follow the alternative with known transition probabilities, we get no new information about the unknown probabilities, and therefore there will never be any reason to change our minds about following the unknown alternative, that is, once we start to follow alternative 1 we follow it forever. Thus, the appropriate expression to enter for the choice of alternative 1 in our expression, is simply the present value of the income we would get by following that alternative forever. This is denoted by $v_2 (P^1)$, and is computed from the equations of Chapter I.

We can also write an expression for V_1^N (r, n) in terms oi V_2^N (r, n) so that our expression involves only one unknown. Since state 1 has known probabilities,

$$V_1^N(r,n) = \frac{p_{11}\, r_{11} + p_{12}\,[\, r_{12} + \beta\ V_2^N(r,n)\,]}{(1 - \beta\, p_{11})}$$

Moreover, in the equation for $V_2^N(r,n)$, the superscripts are superfluous, since our definition of the decision stage forces N+n to be a constant. Thus, the fact that the right side is superscripted (N - 1) is already conveyed by the fact that the argument is (n+1). So, we can drop the superscripts and write our dynamic programming equations as:

$$V_2(r,n) = \text{Max}\left\{ v_2(P^1),\ \left[\frac{r}{n}\left(a_1 + a_2\ V_2(r+1,\ n+1)\right)\right.\right.$$
$$\left.\left. + \left(1 - \frac{r}{n}\right)\left(a_3 + a_4\ V_2(r, n+1)\right)\right]\right\}$$

where again $v_2(P^1)$ is the value of following the known alternative 1, and:

$$a_1 = r^2_{21} + \frac{\beta}{(1 - \beta\, p_{11})}\ (p_{11}\, r_{11} + p_{12}\, r_{12})$$

$$a_2 = \frac{p_{12}\ \beta^2}{1 - \beta\ p_{11}}$$

$$a_3 = r^1_{22}$$

$$a_4 = \beta$$

3.3 Knowledge Space

The function $V_2(r,n)$ is defined over a two dimensional space, and may be plotted in three dimensions. However, what is critical for decision purposes is the plot of the points where the two expressions in the maximum above are equal. On one side of this boundary, we will always choose alternative one, and on the other side we will always choose alternative two. Thus, given this boundary, and knowing which side of it corresponds to alternative one, our decision problem is solved. Given any state of knowledge (r,n) we merely look at our graph and see which alternative to follow. In the next sections, we will give a method for computing this boundary, but let us first pause to investigate which side of the boundary corresponds to which policy.

Under certainty, the value as a function of the transition probability in state 2, p, can be expressed as:

$$V_2(p) = \frac{a_1 p + a_3 (1 - p)}{1 - a_2 p - a_4 (1 - p)}$$

where the a_i are, as defined in the last section. Then:

$$\frac{d V_2}{d p} = \frac{(a_1 - a_3)(1 - a_4) - a_3 (a_4 - a_2)}{[1 - a_2 p - a_4 (1 - p)]^2}$$

The denominator of this expression is always positive, so the derivative takes the sign of the numerator, which is a constant. That is, the value function monotonically increases or decreases as a function of p. Thus, the policy with known transition probabilities may be above or below the boundary, but this may be determined by inspecting the sign of the constant numerator above for any problem. If the derivative is positive the "known" policy would be followed when we had states of knowledge lying above the decision boundary.

3.4 Numerical Computation Using Dynamic Programming

Suppose that for some N and for all r we know the values of $V_2(r, N)$. Inspection of the dynamic programming equations then reveals that we can compute the value of $V_2(r, N-1)$ from these for all r, and so on down to $V_2(r, 0)$. This can be done by simple calculation since the entire right hand side of the equation is constant except for values of $V_2(r, N)$ which we assume we know. Thus, the entire problem reduces to finding the values along some value n. If we were approximating the infinite case by a finite duration case, we could just assign termination values at some large value of n and iterate back, but finding the termination values discussed in Section 3.2.4 is itself a very difficult problem. A much simpler approximation is to say that for some large value of n, say n=1000, we (virtually) know the value of p with certainty as being $\frac{r}{n}$. Thus,

$$V_2(r, N) = \text{Max}\ \{ v_2(P^1), v_2(P^2) \}$$

where P^2 is the known matrix with $p = r/N$.

We have used this computational device to evaluate the decision boundaries. The use of discounting assures us that if we take N large, the terminal values assigned there will not make very much difference in any case, and experience has shown that values of N around 50 yields results almost identical to values around 1000.

3.5 Computation Results

A compute r program was written to find the solution to this two state problem. Its specific objectives were to illustrate the shape of the decision boundary, the speed of convergence with N (the assumed "infinity"), the sensitivity of the decision boundary to the discount factor, and the shape of the isovalue curves in the (r, n) plane.

Several problems were run with different values of β, P_2 and N to illustrate these various properties.

For the first 6 problems the decision boundary points are plotted in the (r, n) space (see graphs following). These are the points where it is f'rst best to pick the alternative 1 when in state 2. ɔr instance, in the second problem (see graph 2), if we had n = 10, r = 8, we would deᴗide to follow policy 2 in the next transition. However, if n = 10, r = 9, we would follow policy 1 in the next transition (and hence, ˉɔrever, since r and n will not change). The problems and parameters are:

1) $\beta = .9,\ N = 50,\ P_{11} = \frac{3}{4},\ P^{1}_{22} = \frac{1}{2}$

2) $\beta = .9999,\ N = 50,\ P_{11} = \frac{3}{4},\ P^{1}_{22} = \frac{1}{2}$

3) $\beta = .9,\ N = 50,\ P_{11} = \frac{3}{4},\ P^{1}_{22} = \frac{3}{4}$

4) $\beta = .9999,\ N = 50,\ P_{11} = \frac{3}{4},\ P^{1}_{22} = \frac{3}{4}$

5; $\beta = .9.\ N = 50,\ P_{11} = \frac{3}{4},\ P^{1}_{22} = \frac{1}{4}$

6) $\beta = .9,\ N = 1000,\ P_{11} = \frac{3}{4},\ P^{1}_{22} = \frac{1}{2}$

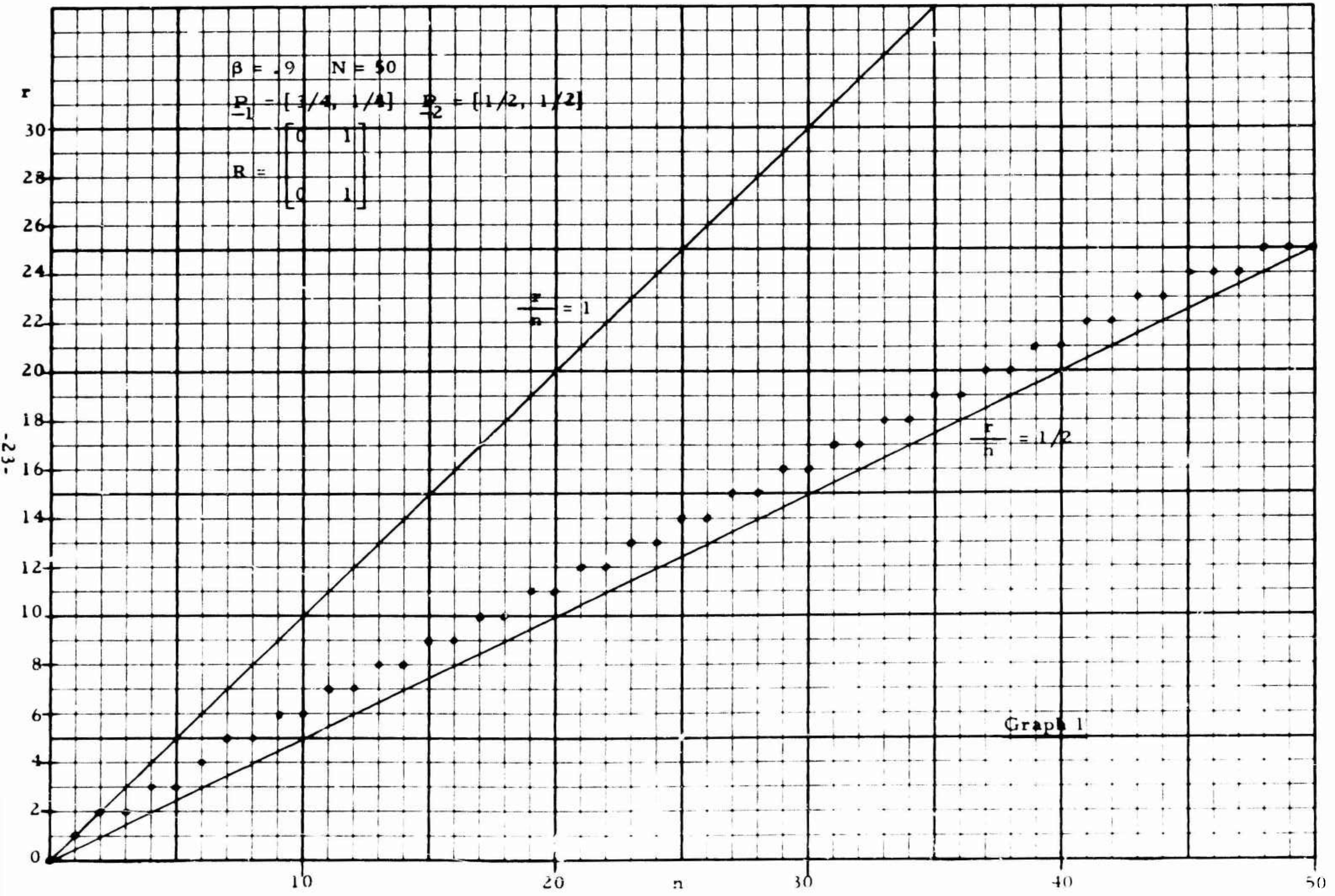
The purpose of the first 6 cases is to illustrate the shape of the decision boundary and how it changes with the value of the discount factor β. Notice that for small values of β the decision points stay very close to the boundary under certainty (straight liₗe below the points). Only for larger β, where the future has great importance, does the boundary move away from the certainty case. In the limit (as β approaches one) the boundary approaches the 45^{0} line--experiment forever.

Comparison of Figure 1 and Figure 6 illustrates the speed of convergence with N for β = .9. Figure 6 also shows the asymptotic behavior of the decision boundary. Notice that the decision points remain the same for N = 1000 and N = 50 except for those near 50. For N = 1000 the decision boundary remains parallel to the certainty boundary for as far out as we have investigated.

The next series of graphs is concerned with a continuous decision boundary and isovalue curves obtained by linear interpolation. The significance of interpolated values is that if the value function V (r, n) is defined for all r and n, and is relatively smooth and flat, then the values can be approximated by interpolation. Graph #7 shows V (r, n) plotted against r for several values of n. Apparently the value function is indeed smooth and flat enough to justify interpolation between the values on the lattice. However, since at the assumed horizon n is an integer (= 50) and r is also taken to be an integer for decision purposes; the results that follow do not have much operational meaning, and are helpful only to get a rough idea of the bahavior of the value V_2 (r, n).

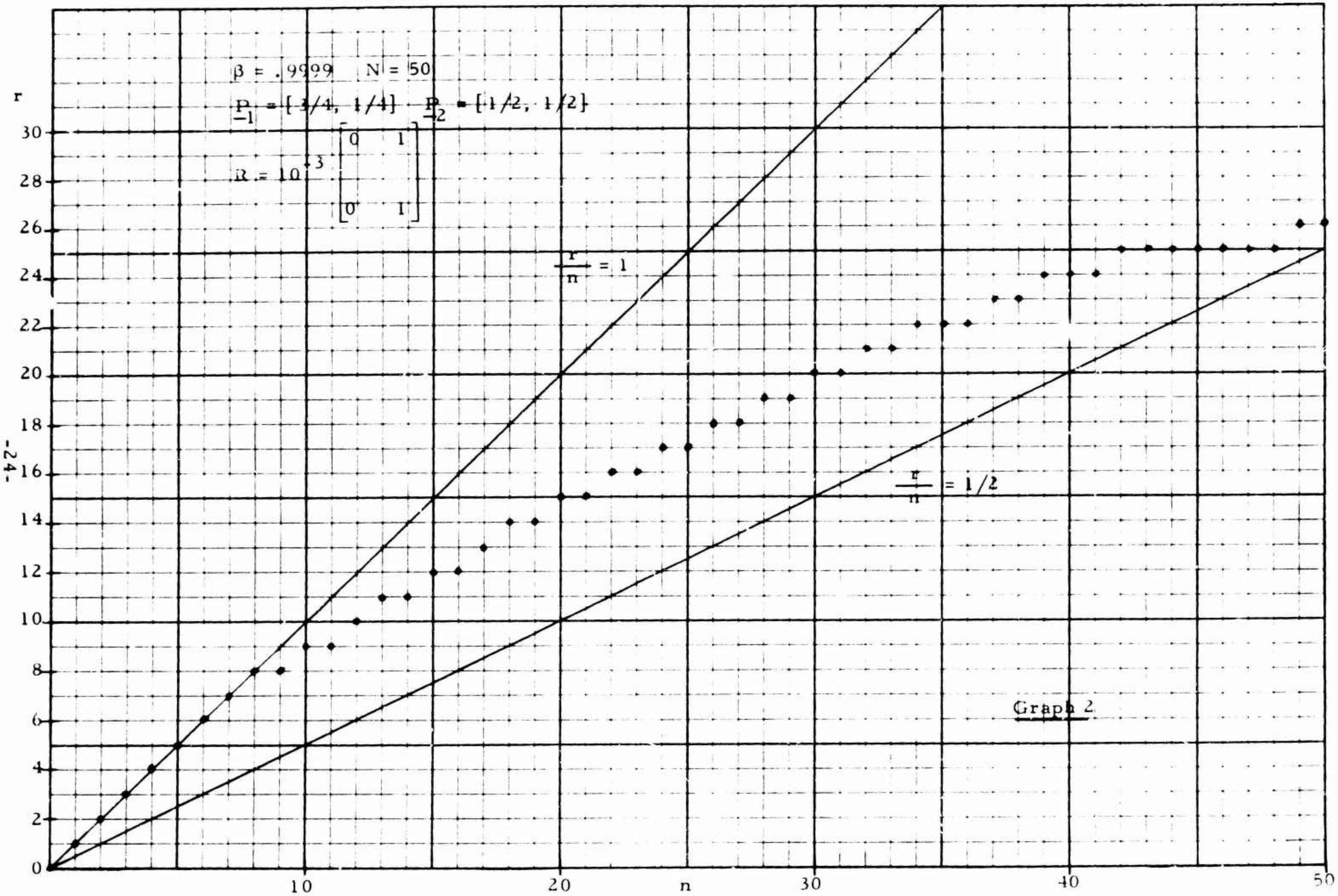
The next graph, #8, shows the decision boundary found by interpolation for three values of β. The significant feature of these curves is their "lumpy" nature. The end points of the lumps occur at integers, where decisions could be made.

The next two graphs show isovalue curves. Isovalue curves which are for values of V_2 (r, n) close to the n-axis appear to be straight lines and have a slope which is very close to the slope that the value under certainty function has. However, they do not pass through the origin when extrapolated. Rather, they have a small positive intercept. For isovalue curves with values close to the certainty boundary, though, the lumpy appearance shows up again. We have not been able to provide a satisfactory explanation for this phenomenon.

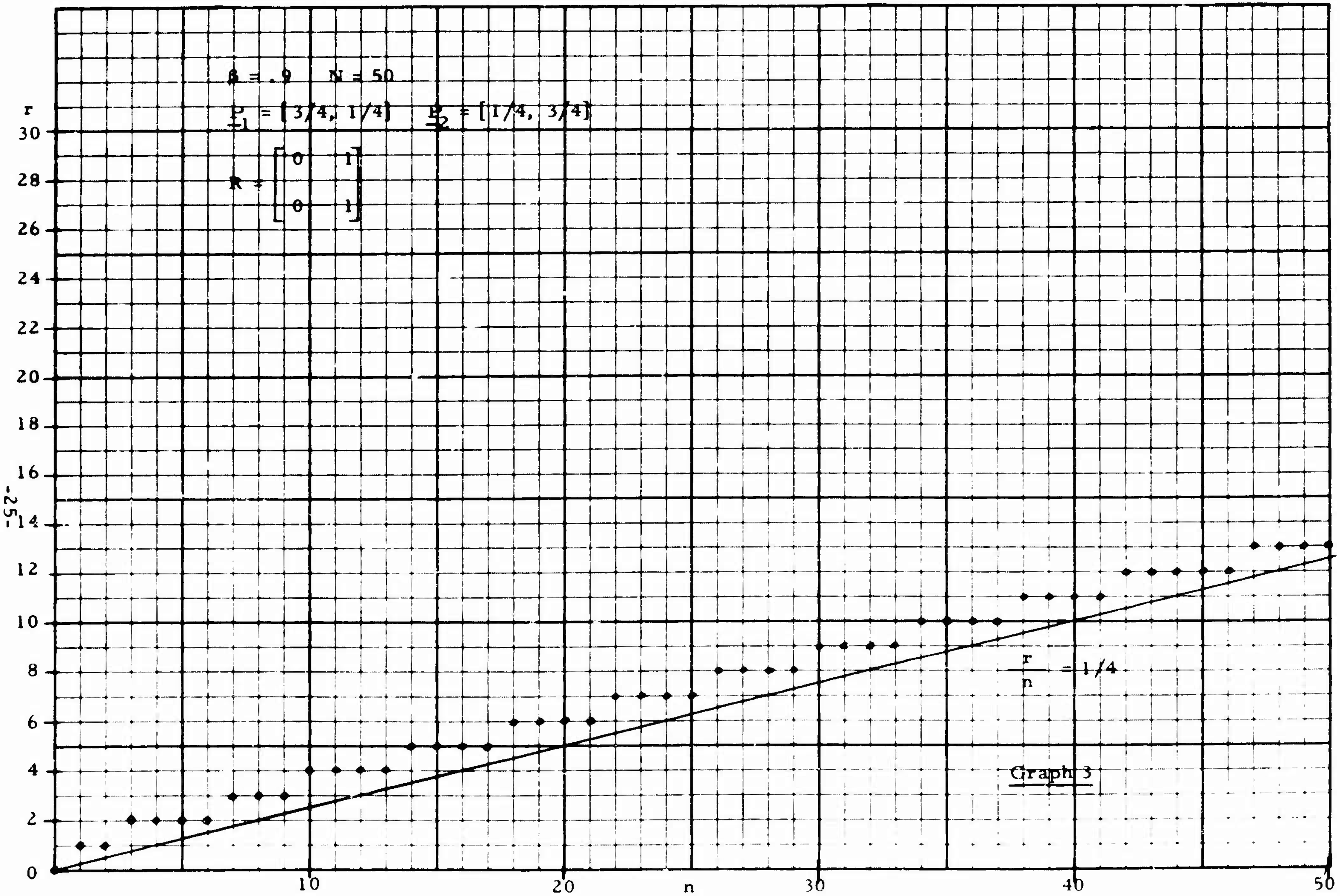


$\beta = .9 \quad N = 50$
$\underline{P}_1 = [3/4, 1/4] \quad \underline{P}_2 = [1/2, 1/2]$
$R = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$
$\frac{r}{n} = 1$
$\frac{r}{n} = 1/2$
r
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
n
10
20
30
40
50

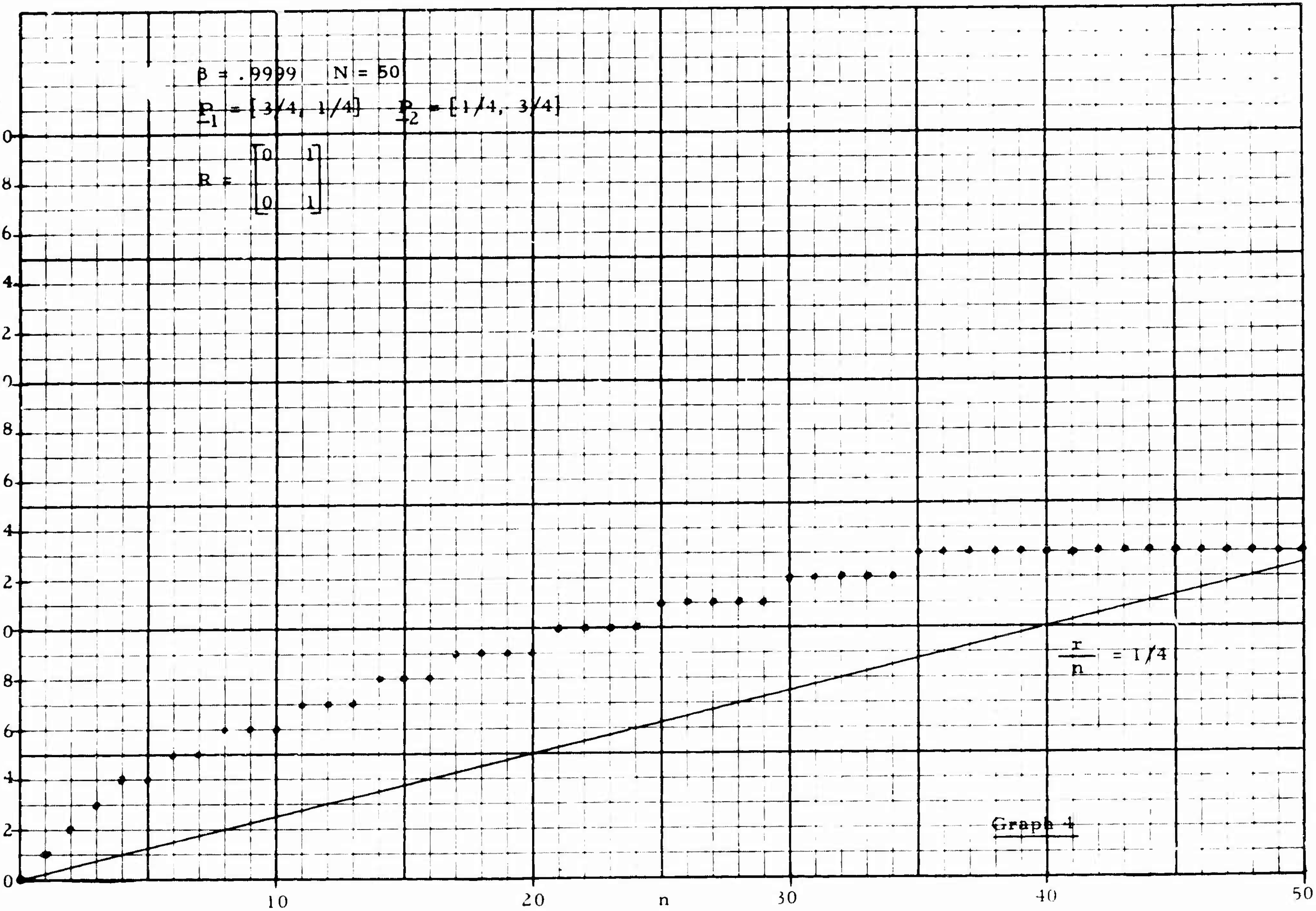
Graph 1


β = .9999 N = 50
P_1 = [3/4, 1/4] P_2 = [1/2, 1/2]
R = 10^{+3} $\begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$
r/n = 1
r/n = 1/2
r
30
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
10
20
n
30
40
50

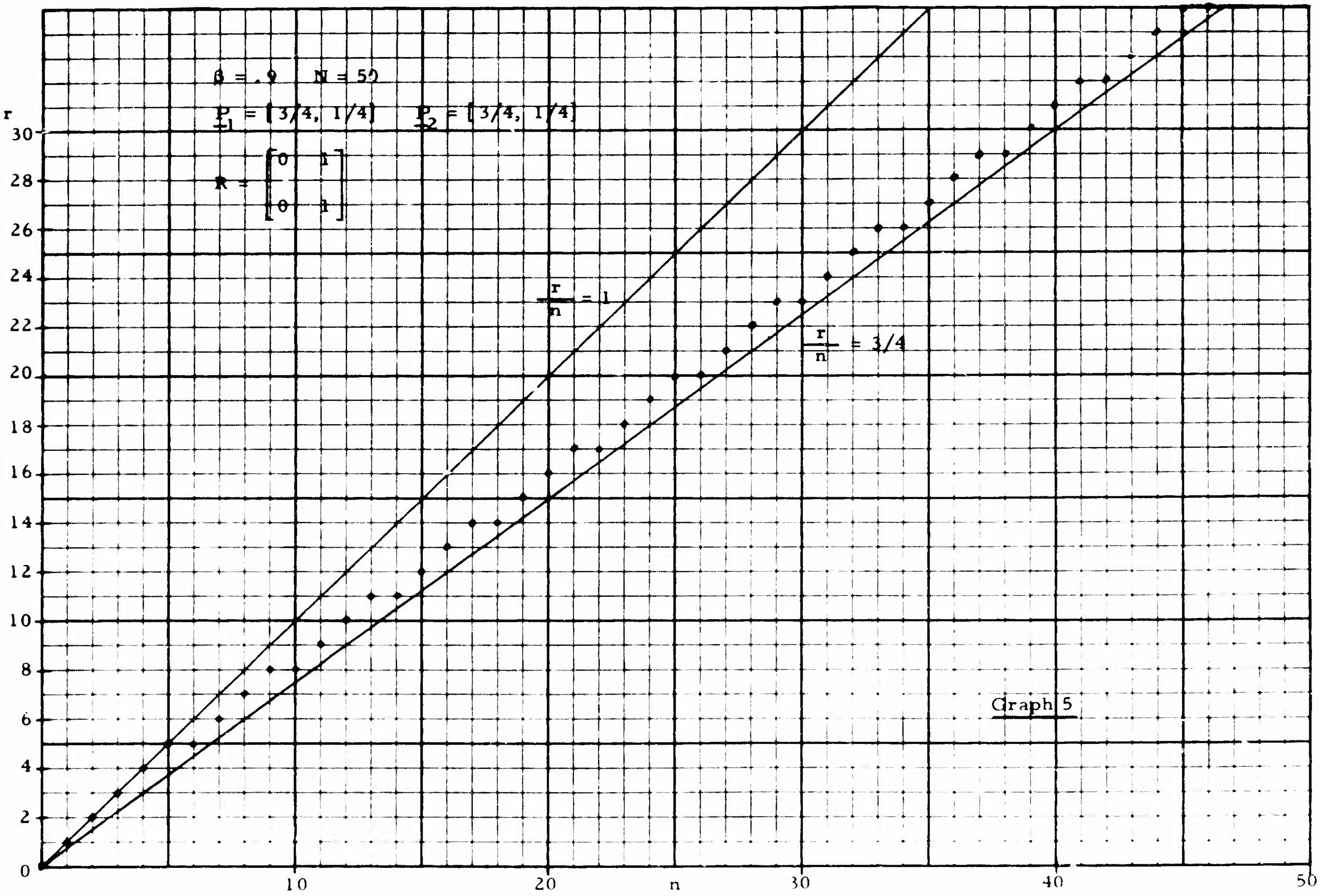
Graph 2


β = .9 N = 50
$\underline{P}_1$ = [3/4, 1/4] $\underline{P}_2$ = [1/4, 3/4]
$R = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$
r
30
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
10
20
n
30
40
50
$\frac{r}{n} = 1/4$

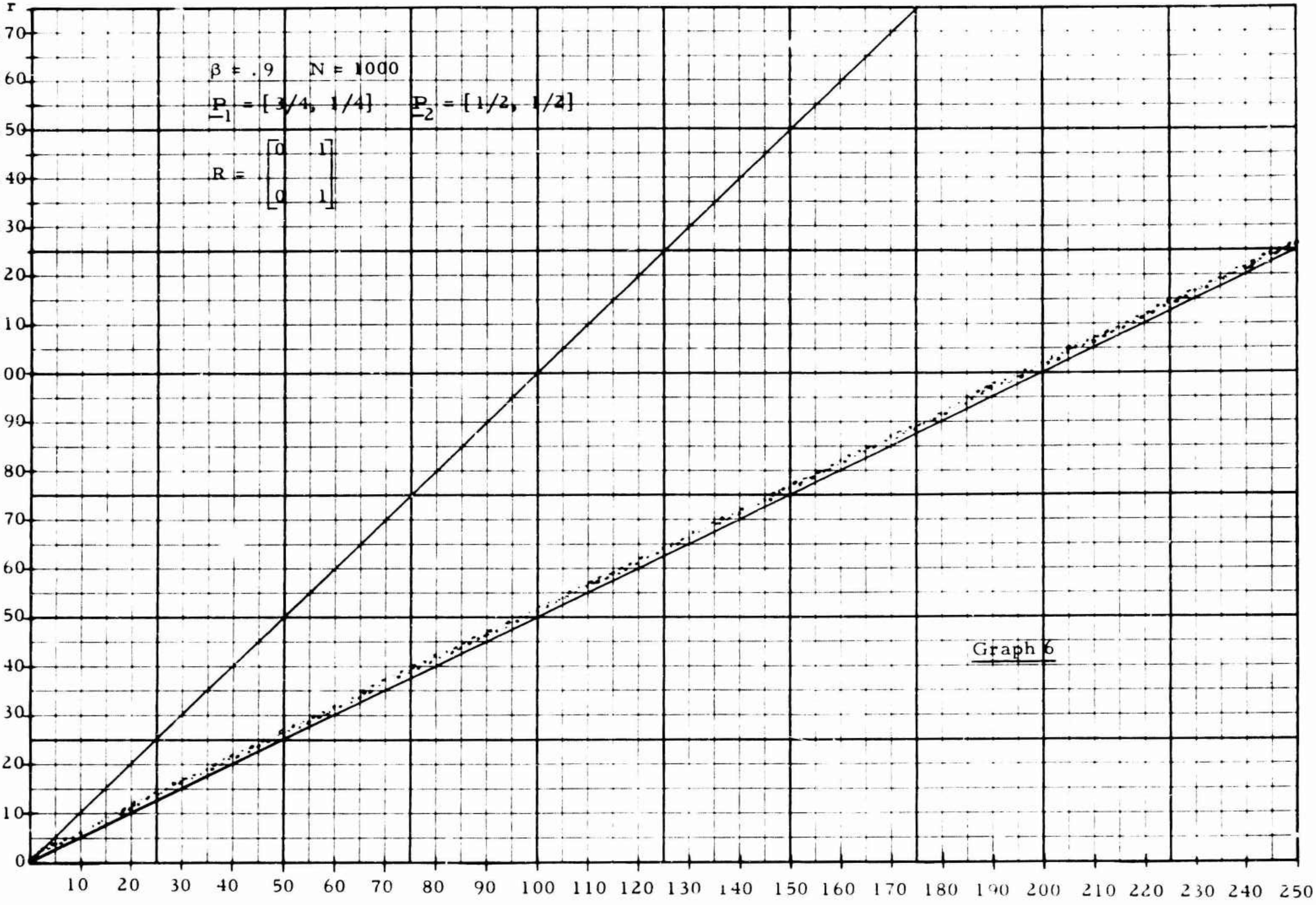
Graph 3


β = .9999 N = 50
$\underline{P}_1$ = [3/4, 1/4] $\underline{P}_2$ = [1/4, 3/4]
$B = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$
r
30
28
26
24
22
20
18
16
14
12
10
8
6
4
2
0
10
20
n
30
40
50
$\frac{r}{n} = 1/4$

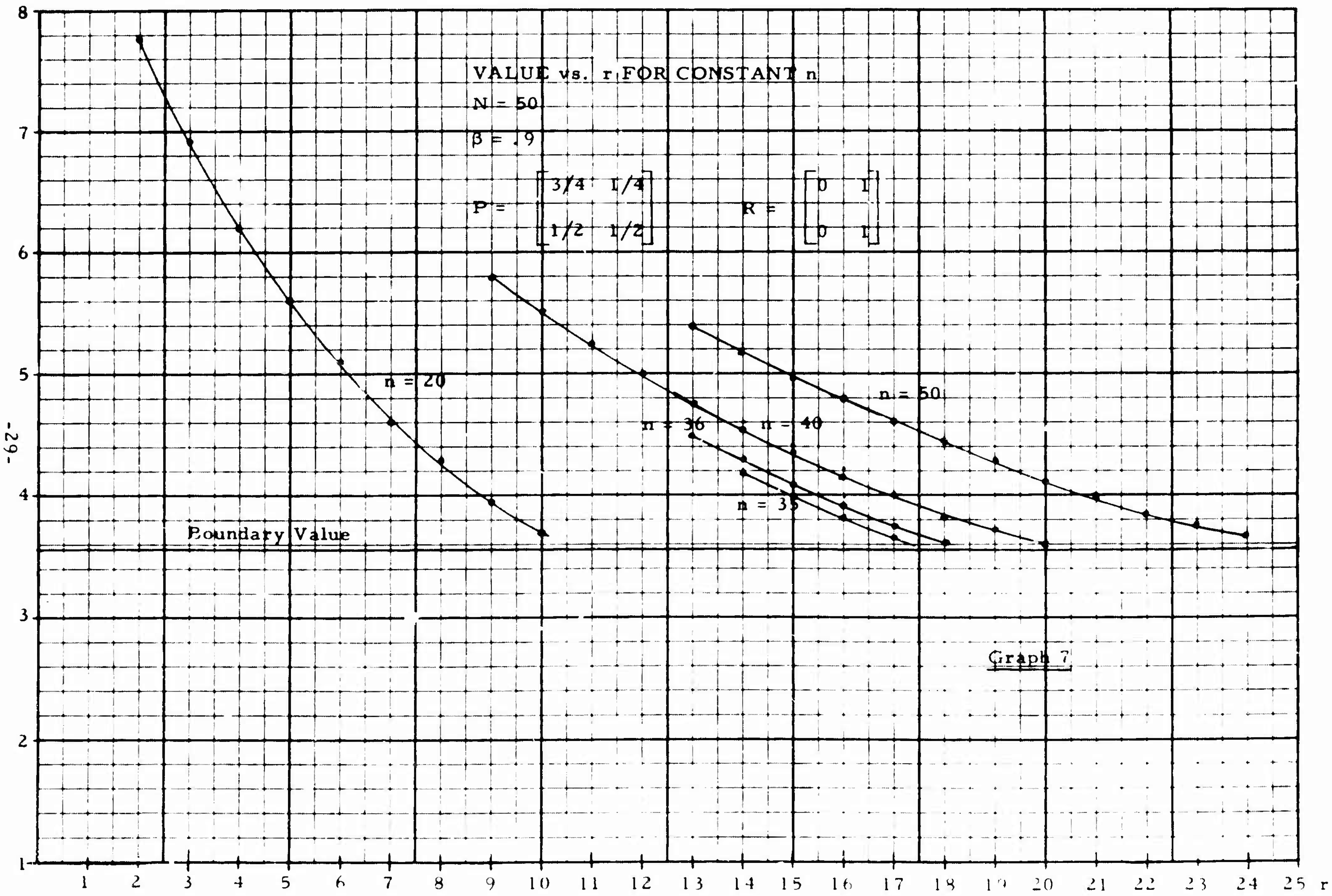
Graph 4


β = .9 N = 50
$\underline{P}_1 = [3/4, 1/4]$ $\underline{P}_2 = [3/4, 1/4]$
$\underline{R} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$
$\frac{r}{n} = 1$
$\frac{r}{n} = 3/4$
r
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
10
20
n
30
40
50

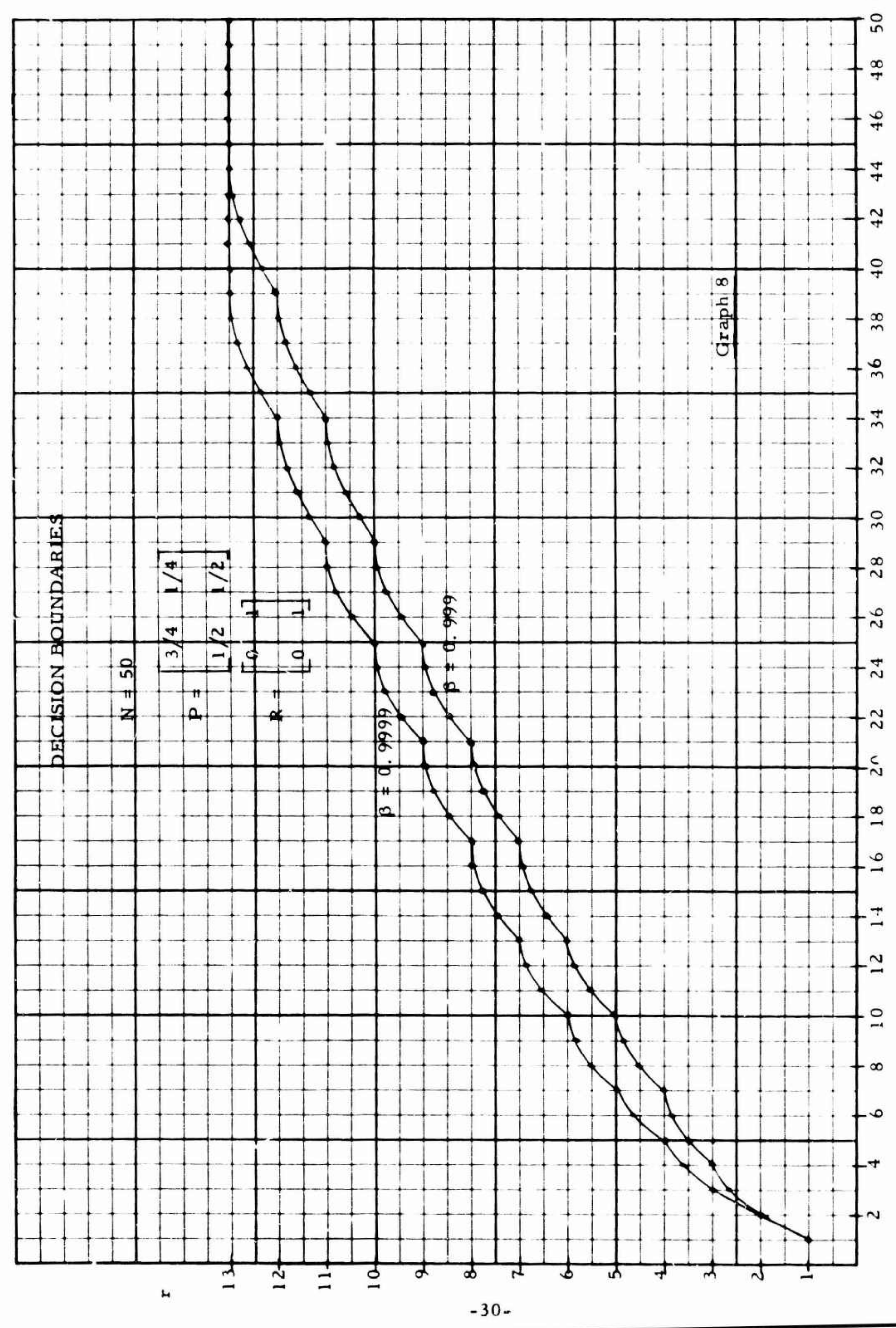
Graph 5


r
170
160
150
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150
160
170
180
190
200
210
220
230
240
250
$\beta = .9 \quad N = 1000$
$\underline{P}_1 = [3/4, 1/4] \quad \underline{P}_2 = [1/2, 1/2]$
$R = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$

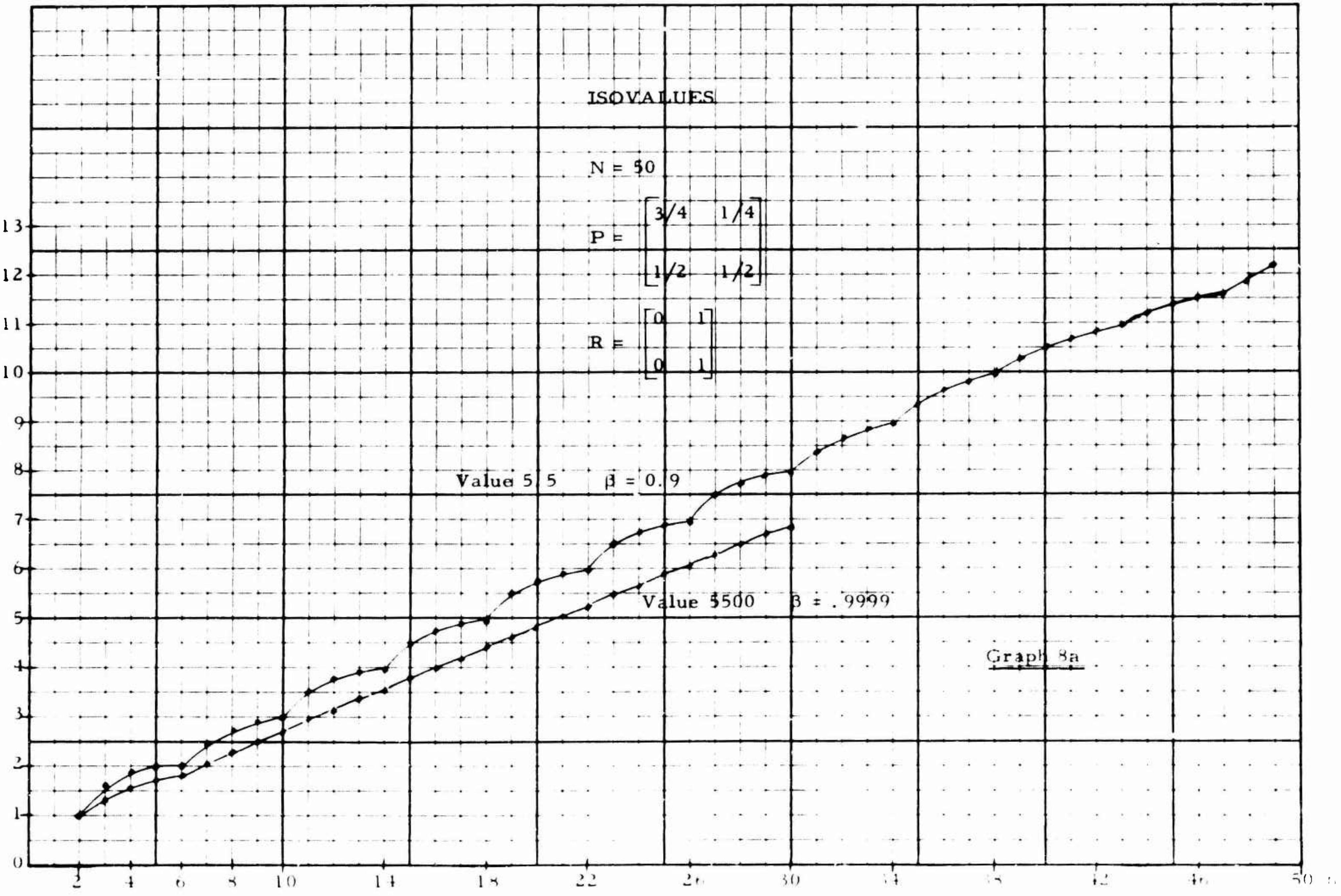
Graph 6


VALUE vs. r FOR CONSTANT n
N = 50
β = .9
P = [3/4 1/4; 1/2 1/2]
R = [0 1; 0 1]
n = 20
n = 36
n = 40
n = 50
n = 35
Boundary Value
8
7
6
5
4
3
2
1
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 r

Graph 7


DECISION BOUNDARIES
N = 50
P = $\begin{bmatrix} 3/4 & 1/4 \\ 1/2 & 1/2 \end{bmatrix}$
R = $\begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$
β = 0.9999
β = 0.999
r
1
2
3
4
5
6
7
8
9
10
11
12
13
n
2
4
6
8
10
12
14
16
18
20
22
24
26
28
30
32
34
36
38
40
42
44
46
48
50

Graph 8


ISOVALUES
N = 50
P = $\begin{bmatrix} 3/4 & 1/4 \\ 1/2 & 1/2 \end{bmatrix}$
R = $\begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$
Value 5.5 β = 0.9
Value 5500 β = .9999
0
1
2
3
4
5
6
7
8
9
10
11
12
13
2
4
6
8
10
14
18
22
26
30
34
38
42
46
50

Graph 8a

3.6 Conclusion

In this chapter we have considered the exact solution to a simple two state case. We have shown the dynamic programming method of solution, and the numerical solutions for several values of the problem parameters. The conclusions drawn from this work are:

1) When $\beta = .9$ or smaller, the value under certainty gives a good approximation to the value under uncertainty for n greater than 15 or 20. The isovalue lines are displaced upward slightly from the isovalue lines of the certainty case, but their slope is undisturbed.

2) As β is increased from .9, the effects of uncertainty are more pronounced. The decision boundary bulges upward and becomes lumpy. Isovalue lines near the boundary are similarly affected, though those far from the certainty boundary remain remarkably straight lines.

Now that we have considered the nature of an exact solution for a simple case, we will discuss in the next section the difficulties encountered in trying to apply this method to even a slightly larger problem.

CHAPTER IV

INFEASIBILITY OF A DYNAMIC PROGRAMMING SOLUTION FOR AN N-STATE PROCESS

4.1 Terminal Analysis

We demonstrate the infeasibility of dynamic programming by formulating the equations for the general case, and showing that their solution requires a prohibitive amount of calculation.

As in the special two state example presented in the last chapter, we approximate the infinite case by a long finite process with terminal rewards. We calculate the terminal rewards by assuming that we must choose a policy at the 0^{th} stage which we will follow forever.

Exactly as in the two state case we define:

$$V_i^L \,(m_{11}^1,\ m_{11}^2, \ldots m_{11}^{k_i};\ m_{12}^1 \ldots m_{ij}^k, \ldots\ldots m_{NN}^{k_N})$$

as the present discounted value of the infinite reward stream under the optimal strategy given that we are now in state i with L decisions left before the terminal decision. The decision stage is now defined as just the period (holding time) of the process, since a decision must in general be made in each state.

This value is obviously a function of all our present knowledge, so all prior parameters appear. For simplicity, we denote the matrix of prior parameters by M, and write $V_i^L\,(M)$.

Finally, at the terminal stage we assign a policy, and the present discounted value of a policy given the transition probabilities can be computed by the formulas of Chapter I. So:

$$V_i^0\,(M) = \underset{\text{all policies } A}{\underline{\text{Max}}} \left\{ \int_{P^A} v_i\,(P^A)\, f_\beta\,(P^A \mid M) \right\}$$

where, as usual, $v_i(P^A)$ is the present discounted value of being in state i under policy A when P^A is known.

4.2 The Recursive Equations

If we are now in state i and following alternative k with L stages left before the terminal decisions, we can go to any state j with probability $\tilde{p}_{ij}^k$. If we go to j, we earn r_{ij} immediately, but as of next period we can expect $V_j^{L-1}(M')$, where M' is the matrix of posterior parameters given the transition from i to j. Let I_{ij}^k denote a matrix of zeros except for a single one in the position corresponding to m_{ij}^k. Then:

$$V_i^L(M) = \underset{k}{\text{Max}} \left\{ E\left[\sum_{j=1}^{N} \tilde{p}_{ij}^k \left[r_{ij}^k + \beta V_j^{L-1}(M+I_{ij}^k) \right] \right] \right\}$$

This expectation is seen to be linear in $\tilde{p}_{ij}^k$, so we can just use the statistics of the expected process. Thus:

$$V_i^L(M) = \underset{k}{\text{Max}} \left\{ q_i^k + \sum_{j=1}^{N} \beta\, p_{ij}^k\, V_j^{L-1}(M+I_{ij}^k) \right\}$$

These equations closely resemble the ordinary equations for determining an optimal policy with known transition probabilities, except that now $V_i(\cdot)$ has different arguments on the two sides of the equations.

4.3 Conclusions

There are difficulties inherent in solving both the terminal and the recursive dynamic programming equations. The form:

$$V_i^0(M) = \underset{A}{\text{Max}}\ E\left[v_i(\tilde{P}^A) \mid M \right]$$

is itself virtually intractable, as it involves a very complex multiple integration. But, as will be discussed in the next chapter, it may be appropriate to say that:

$$E\,[\,v_i\,(\widetilde{P}^A)\;|\;M\,] \cong v_i\,(P^A)$$

where P^A is the matrix of policy A in the expected process. In this case, the terminal equations could in fact be solved by ordinary policy iteration.

Given $V_i^0\,(M)$ for all values of M, there is no theoretical difficulty in solving the rest of the problem. The only trouble is one of dimensionality. M is a $(\sum_{j=1}^{N} k_j \times N)$ matrix, and the state superscript i can run from 1 to N. If we set even so modest a task as to tabulate for each integral m_{ij}^k in the interval (1, 100), we find that each decision stage involves:

$$N \cdot (100)^{\sum_j Nk_j}$$

tabulations. Even for very small problems this is impossible. For example, our taxicab example would require 3×100^{24} tabulations for each decision stage. To reach convergence would require at least 100 stages, so 3×10^{50} calculations would be required.

But even though a dynamic programming solution is infeasible, it does provide us with a compact formulation of the problem, and a clear notion of what a solution is. If we let L grow very large so that L becomes indistinguishable from (L 1), then the problem can be stated.

For every possible prior matrix M and every state i, specify that alternative k^* such that:

$$V_i(M) = \underset{k}{\text{Max}} \left[q_i^k + \beta \sum_{j=1}^{N} p_{ij}^k \cdot V_j(M+I_{ij}^k) \right]$$

CHAPTER V

DISTRIBUTION OF LONG RUN AVERAGE GAIN

5.1 Steady state probabilities

One measure of effectiveness for a Markov process without discounting is the long-run average gain, which is defined as the sum of the products of the immediate rewards by the corresponding steady state probabilities:

$$g = \sum_i \pi_i q_i$$

Both the q's and the π's depend on the transition probabilities, so that in the case we are studying, they are both random variables.

E. A. Silver has shown that the $E(\tilde{\pi}_j)$'s can be approximated by the π_j's of the expected process when the transition probabilities are multidimensional beta distributed.[3] The accuracy of this approximation was shown by using simulation techniques. His results also indicated that the approximation became better as the N_i's increased.

This result sheds partial light upon our problem, but does not go quite far enough. Our primary interest is in the product of the $\tilde{\pi}$'s with the $\tilde{q}$'s, and it is the behavior of this statistic that we shall study in the following sections.

5.2 Distrib ion of gain for a given policy

In order to know how bad it might be not to follow the optimal policy, it is useful to have some idea of:

a) What the distribution of gain is like for a given policy, and

b) How that distribution differs for various policies?

The second question could be answered by the first if we found the distribution of gain explicitly in terms of the prior parameters of the policy.

It is necessary to point out that throughout this chapter we will be dealing with terminal analysis; that is, the same policy will be followed forever. This is a first step towards the analysis of the general problem which allows the policy to be changed at any time.

5.2.1 Simulation runs

In order to study the distribution of the gain, a sampling program was written. This program receives as data the prior parameters of a Markov process with one alternative in each state (thus there is only one policy; a necessary restriction here since gain is associated with a particular policy). The program produces sample processes from the prior parameters and computes the gain of each sample process. The range of the gain is divided into several intervals and for each interval a count is kept of the sample gains in that range. This provides a histogram of the distribution. Besides, a sample mean and variance is computed. (See Appendix A.14)

We have studied five particular processes. The first is a five-state process with probabilities and rewards drawn from a random number table. The other four were selected policies of the three-state taxicab problem.

Each time a process was studied, the prior parameters N_i were the same for all rows (states). For different computer runs, different values for the N_i were selected. Thus, the behavior of each process was studied for four different values of the N_i: 25, 50, 100, and 150. There were also three miscellaneous runs. One was the five-state problem but the rewards (which had a range of zero to ten) were replaced by their ten's complement. The second miscellaneous process was the regular five-state problem, but with different values of N_i for the various states (the average value was 100). The last run was policy (2, 2, 2) of the taxicab problem, but only 25 sample processes were drawn to see how accurate statistics could be obtained by very little sampling. There were 23 runs in all.

5.2.2 Results of the simulation

The sampling experiments described above yielded only limited evidence of the system behavior. The experiments were in most respects exploratory rather than aimed at testing any one particular hypothesis. Even so, there are certain results that seem significant enough to be mentioned here. The following conclusions are substantiated by the accompanying graphs and tables.

E. A. Silver has shown that the general form for the π_j's involves N^{N-1} cross product terms in the numerator (for an N state process), and N^N cross product terms in the denominator. We originally hoped that for large processes, the law of large numbers might come into play, and that the distribution of the gain would be approximately normal. This was one of the reasons we tabulated a sample mean and variance when plotting our histogram.

a.) When the distribution of the gain is plotted on probability paper, it does approximate the normal for larger processes. Also, the larger the value of N_i, the more normal the distribution appears. The mean and variance of the normal are very close to the sample mean and variance computed. But, however encouraging this result, it should be mentioned that a chi-square analysis shows quite decidedly that the distribution is not normal.

In other words, the normal is clearly only an approximation. Since the N^N terms mentioned above are not independent, the law of large numbers need not apply, even for very large processes.

b.) For a given process, the product of the sample variance and the common value of the N_i's appears to be more or less constant. This suggests that given the constant (which is seen to vary from process to process), we could predict the variance merely by knowing the prior parameters N_i when they are the same in every state. Our miscellaneous run indicates that the statistics do not change much if the N_i are slightly different--we can just use their mean value.

c.) The mean value of the gain is approximated well by the gain of the expected process, and this approximation becomes better and better as the prior parameters N_i increase. This is an extension of E.A. Silver's result that the above is true for the π_j alone.

d.) Sometimes the gain of the expected process is larger than the sample mean, and other times it is smaller. The two cases are illustrated by the case where we took 10's complement of all rewards. We have determined no a priori way of determining which case will pertain in a particular process.

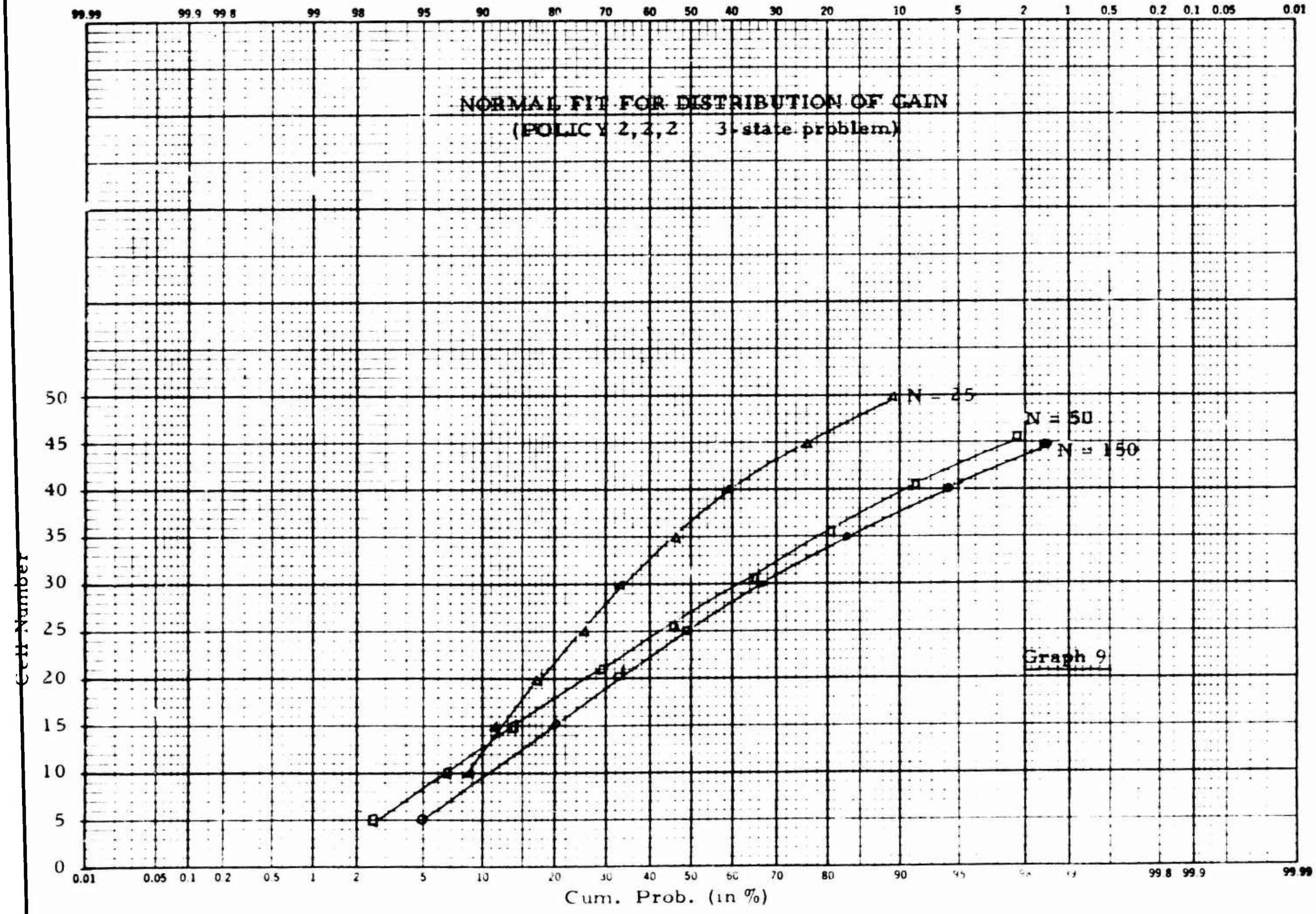
RESULTS OF SIMULATION OF TAXICAB EXAMPLE

500 sample processes for each simulation

Policy	N	g	$E(\tilde{g})$	$\mathrm{Var}(\tilde{g})$	$N \cdot \mathrm{Var}(\tilde{g})$
2, 2, 2	25	13.34	14.04	1.3047	32.5
	50		13.42	.6399	32.0
	100		13.39	.3401	34.0
	150		13.44	.2458	36.8
1, 2, 2	25	13.15	14.04	1.1375	28.4
	50		13.31	.6784	33.9
	100		13.23	.3098	31.0
	150		13.32	.2117	30.4
2, 1, 2	25	8.81	8.67	.1107	2.27
	50		8.72	.0504	2.52
	100		8.79	.0228	2.28
	150		8.79	.0201	3.01 (only 62 samples)
2, 2, 1	25	12.89	13.93	1.2716	31.8
	50		12.96	.7494	37.5
	100		12.93	.4604	46.0
	150		12.92	.2688	40.3
2, 2, 2	25	13.34	13.72	1.5228	(25 samples)

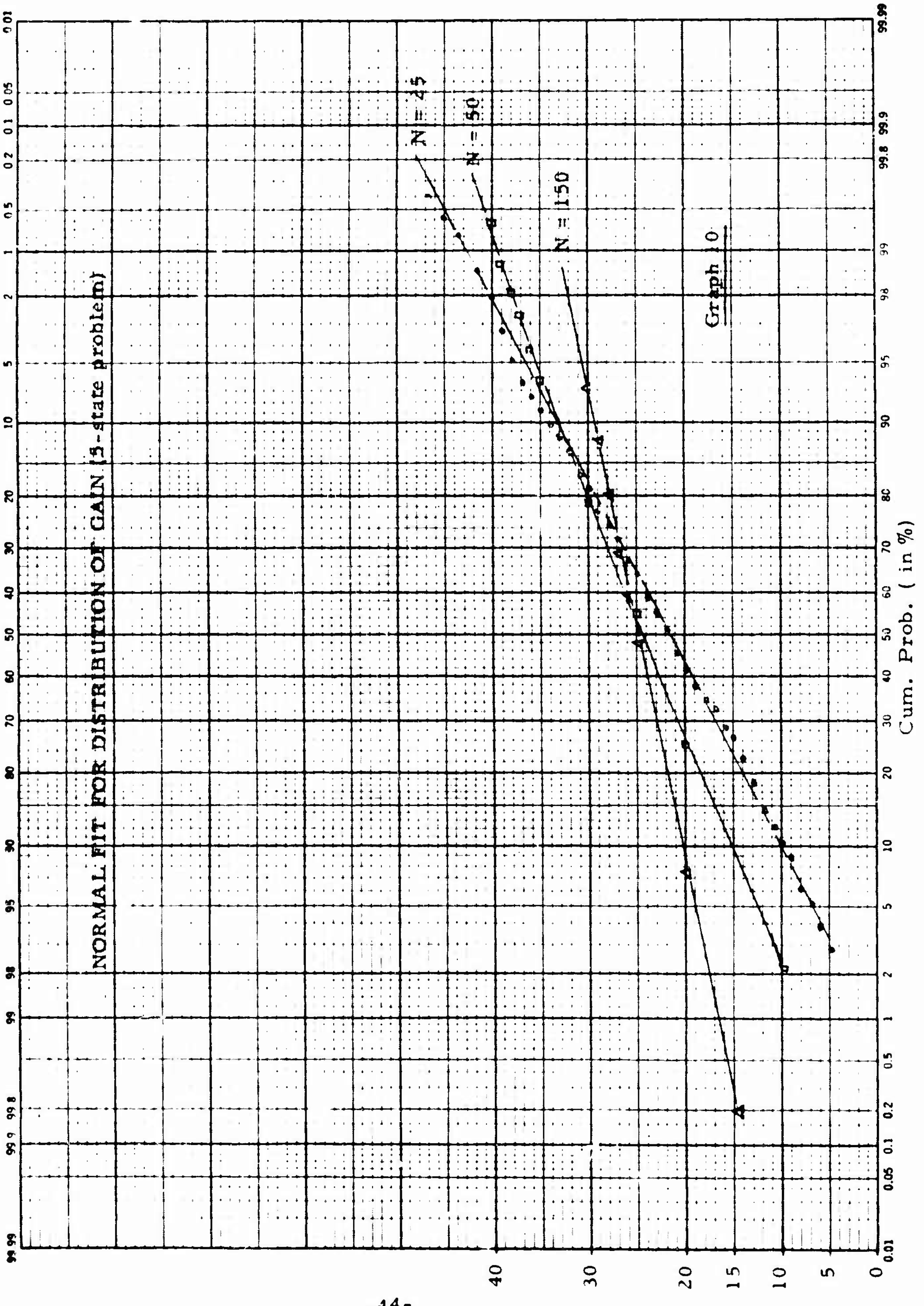
RESULTS OF SIMULATION ON FIVE-STATE RANDOM PROCESS

Number of Samples	N	g	$E(\tilde{g})$	$Var(\tilde{g})$	$N \cdot Var(\tilde{g})$	
500	25	4.69	4.60	.0683	1.7075	$g \geq E(\tilde{g})$
455	50		4.67	.0365	1.825	
220	100		4.68	.0167	1.670	
500	150		4.69	.0096	1.440	
500	different ave = 100		4.68	.0155		
		"Reversed rewards" case				
500	150	5.38	5.39	.0091		$g < E(\tilde{g})$



NORMAL FIT FOR DISTRIBUTION OF GAIN
(POLICY 2,2,2   3-state problem)

Graph 9



NORMAL FIT FOR DISTRIBUTION OF GAIN (5-state problem)

Graph 10

5.3 Limitations and suggestions for future work

In concluding this chapter we would like to point out some of the limitations of our research into the distribution of the long run average gain.

In the first place, we have studied a very limited number of processes (only five), and they have been of small size (three-state and five-state). It would be quite desirable to study nore problems of larger size to test the generality of our results.

Second, in all the processes studied (except for one) all the N_i's had the same value, and only four different values were considered. A few processes with larger N_i's should be studied, as well as more processes where the N_i's for the same process have different values in different states.

Third, we did not study the effect of different reward structures on the same transition probability structure. Whether the two can be separated is a question which should be studied.

Fourth, all the processes studied in this section were processes with only one alternative in each state. It would be interesting to investigate the general case with alternatives. There would be two ways to approach this. The first would be to take some selected policies and study the distribution of the gain for each one separately. (To some extent, we did this with the taxicab problem.) Another way would be to take the prior parameters for the entire multi-alternative process and to sample complete multi-alternative processes from this. Each sample process could then be solved for the optimal policy, and sample optimal gains could be recorded (and perhaps also compared against the sample gains of some fixed policies). A good number of sample processes would be needed for this program--many times the number of policies (which is itself a very large number).

Fifth, the processes studied have always been completely ergodic. It should be interesting to study the behavior of processes with transient states, and/or multiple chains.

Sixth, we mentioned before that we could predict the variance for larger values of N_i, given the "constant" product of some N_i and its variance. We should note, however, that in actual processes when N_i changed due to observations, the mean values change also, so that in practice different values of N_i are associated with different expected processes. We have not shown that the "constant" remains constant under such operations, but just what it does do would be an interesting question for study.

5.4 Conclusions

With all the limitations pointed out in the last section, it would be very presumptuous to state any general normative rules. However, it seems safe to say the following:

a.) When the prior parameters N_i are large enough (and 150 seems large enough for five state processes), it is a good approximation to assume that the process is known with certainty and that the probabilities are given by the expected process. The value obtained from the expected process is almost identical to the sample mean obtained by simulation, and the sample variance is almost negligible.

b.) Even when the N_i are quite small (say 25 in the five state case), the gain of the expected process is close (say 1/2 standard deviation) to the sample mean, and the standard deviation is only about 10% of the mean value. Moreover, this figure falls as $N^{-1/2}$.

CHAPTER VI

HEURISTIC METHODS

6.1 General considerations

We have previously discussed the analytic difficulties which lead us to a heuristic approach. In this chapter we consider some approaches to the problem which are empirical or intuitive in nature, present some experimental results, and suggest possible extension and generalizations.

The basis for most of the heuristics is that some sort of trade-off is implicit in our problems. On the one hand is the immediate expected reward to be earned from the process if we follow the optimal policy of the expected process, but on the other hand is the possibility of finding a still better policy by further experimentation with relatively unexplored alternatives. It will be noted that, with the exception of the first and simplest heuristic, all of the suggested approaches have one or more parameters which attempt to measure this trade-off.

6.2 Follow optimal policy for expected process

This is, conceptually, the simplest possible heuristic. After every transition we perform a Bayesian modification of our prior. Then, using the matrix of expected values, we determine the optimal policy to follow for the next transition.

Obviously, a single transition is unlikely to greatly effect the policy decision, and it would be expensive to re-examine the policy every period. So for purposes of experimentation, we chose 50 transitions as a convenient and reasonable period for re-examination.

Since this heuristic seems the one that would naturally be used in the absence of some more sophisticated analysis, it is worthwhile to examine it here. In the first place, the trade-off between immediate gain and information does not exist in this heuristic. There is no mechanism which explicitly forces unexplored policies to be observed in early stages. Therefore, if it should happen that there is some very good policy which a priori seemed quite bad, it is entirely possible that this heuristic will never provide the information needed to recognize the policy as being better than originally thought.

On the other hand, if a policy looks very good a priori, and happens to be not so good after all, the heuristic will quickly reveal this. Indeed, since Bayesian modification of the prior is continually taking place, this not-so-good policy will soon become only second best in the matrix of expected values.

Thus, this first heuristic provides one kind of information, but not another. If a good policy looks bad, we may never find this out. But if a bad policy looks good, this is discovered quickly. In practice, even the first kind of information may be obtained; the original best becomes only second best and experimentation begins with an unexplored policy, which may begin to look better. Moreover, in exploring a policy, say [2, 2, 1] . we are also indirectly exploring policies [2, 1, 1] , [1, 2, 2] , etc.

We expect then, that when the true transition probabilities of the process are likely values of the prior distribution, this heuristic method should perform very well. However, when the true transition probabilities are in fact far from the prior expectations, then the initial policy may well be a poor one, and may even fail to generate the information very quickly to indicate that in fact it is an inferior policy.

This last surmise was verified by deliberately choosing an unlikely sample point from a prior distribution, so that the best policy of the actual process looked quite poor, a priori. The process and prior are displayed as Run 1 in Appendix B. Note that in 15 iterations 50 observations each, the true optimum was never explored or discovered--rather, the same apparent optimum was chosen every time.

What is needed is an exhaustive verification of the first assertion, that in an expected value sense, this heuristic will provide very satisfactory results. Simulation with a large number of sample points from a very large number of priors would be needed to establish the assertion quantitatively.

As an example of the type of simulation needed, and to get some clues as to probable outcomes, we constructed Run 10 (Appendix B 1.2). Here we utilized a prior and rewards drawn from random number tables, and simulated for five sample points. The results were surprisingly good, the optimal policy of the expected process was almost always the optimal policy of the sample process, or so close to it that we are earning 96+% of what we could have earned with perfect information.

This figure seems to be quite satisfactory, especially as it improves with time as our knowledge becomes more complete.

More simulation of this sort should be carried out. But we turned our attention toward heuristics which would meet the challenge of extreme points from the prior.

6.3 A trade-off between immediate reward and information

We will now establish measures for immediate reward and for information, and we will then explain our proposed trade-off heuristic. It has always been noted that the q_i^k can be interpreted as immediate expected rewards. And a rough and ready estimate of information inherent in an alternative is $N_i^k = \sum_j m_{ij}^k$, the larger N_i^k, the more information we have about the alternative.

We now define the quantity $w_i^k = (q_i^k - q_i^{min}) \alpha / N_i^k$. Here α is the trade-off constant. The so defined w_i^k is large when either 1) the relative immediate reward is great, or 2) the current information is scant. Thus, our heuristic is to choose a policy by maximizing w_i^k in each state. We then observe under that policy for a number of transitions NOBS which is proportional to the smallest of the w_i in the policy. That is, if the relative immediate rewards are all very great, or the current information in each state is very small, we observe for a long time: NOBS = $\beta \cdot w_{min}$. This heuristic is displayed as MAIN 2 in Appendix A.

Our experience with this heuristic indicates that the parameters α and β are quite problem-dependent, and that the policies selected are very sensitive to changes in either parameter. For a solution to our sample problem using this heuristic see Runs 2, 3, and 4 in Appendix B.

6.4 Weighted immediate reward technique

In this heuristic, we weight each q_i^k by a factor c_i^k before applying the policy iteration algorithm for finding the optimal policy. The weighting factor is similar to the one used in the previous section. Now:

$$c_i^k = (\alpha\, q_i^k / N_i^k) + 1.$$

Note that as the number of observed transitions becomes very large, the weighting factor c_i^k approaches 1, so eventually we converge to the optimal policy of the expected process.

Having found the optimal policy by this technique, we observe under the determined policy for a number of observations NOBS calculated to bring the c_i^k down as low as the minimum such value in the state. This is done to minimize the chances of simply re-determining the same policy. To be explicit, NOBS for the policy A is chosen to be $\sum_i x_i$, where the x_i satisfy:

$$\min_k c_i^k = \frac{\alpha\, q_i^A}{N_i^A + x_i} \qquad \text{for all } i$$

It may be that $x_i = 0$ for some i, which means that the c_i^k associated with the optimal policy is already the lowest in the state. If a given alternative appears in the optimal policy in spite of having the lowest weighting factor, then it must be a very good policy indeed. So, in this case we set $x_i = \beta \cdot N_i^A$. That is, if this good policy has a lot of information to back it up, we observe for a long time.

Runs 5-9 listed in detail in Appendix B, were made using various values for α and β. In all cases, we eventually converge on the expected optimum, since the weighting factors approach 1. But hopefully, this heuristic will also handle extreme points from the prior. The problem then becomes a speed of convergence to the optimum.

To get a measure of speed of convergence, we define an efficiency by

$$\text{EFF} = \frac{\text{Expected actual earnings}}{\text{Expected optimum earnings}}$$

Plots of efficiency versus accumulated number of observations are given in Appendix B.3 for different values of α and β.

It can be seen from these graphs that convergence is not terribly rapid. If a small discount factor were operative, this convergence could be quite unsatisfactory.

6.5 Weighted variance technique

This heuristic is a direct attempt to measure the trade-off between immediate gains and information. As our measure of information, we turn to an analysis of variance.

We first look for an expression for the expected decrease in the variance of the marginal distribution of a p_{ij} if we make one observation of the process let us denote this quantity by d_{ij}^k. This quantity gives some idea about how much will be learned about a particular transition probability if we observe one transition. But some transitions are more important than others; namely, those with high rewards. (This is only a first approximation, as those which put us in states with high expected returns are also important). Thus, we weight each d_{ij}^k by the corresponding r_{ij}^k to obtain a row sum s_i^k. This we call the "total weighted expected change in variance," and assert that it is a dollar measure of information to be gained.

Next, we ask how we can measure the "cost of obtaining information." Suppose we want to observe a particular alternative k in state i. Let 0 denote the optimal policy for the system, and A denote the best policy which uses alternative k in state i. Then $g^0 - g^A$ provides a measure for the expected loss from experimentation.

Then we can state the working of this heuristic: for our present system state we find the alternative which maximizes s_i^k; that is, the state with the greatest expected information gain. We then find the cost of experimenting by computing the approximate $g^0 - g^A$, and compare a weighted value of s_i^k to this difference to decide whether to use the optimum alternative of the expected process, or whether to experiment to gain information with alternative k.

We have now to define the quantity d_{ij}^k, that is, the expected decrease in variance from one observation. We first recall the Bayesian theorem that

mean of posterior variance + variance of posterior mean = prior variance

prior variance - mean of posterior variance = variance of posterior mean

The left hand side of this equation is precisely what we mean by d_{ij}^k, so that

$$d_{ij}^k = \text{variance of posterior mean}$$

This is a general result, true for any distribution and for any amount of sampling. Let us consider the case at hand, a multidimensional beta and one sample. Then the posterior mean is:

$$\frac{m_{ij}^{k} + r}{N_i^k + 1}$$

where r has a beta-binomial distribution. The required variance is:

$$d_{ij}^{k} = \frac{m_{ij}^{k} \left(N_i^k - m_{ij}^{k}\right)}{\left(N_i^k\right)^2 \left(N_i^k + 1\right)^2}$$

Thus, s_i^k is defined by:

$$s_i^k = \frac{1}{\left(N_i^k\right)^2 \left(N_i^k + 1\right)^2} \sum_{j=1}^{N} m_{ij}^{k} \left(N_i^k - m_{ij}^{k}\right) r_{ij}^{k}$$

Again we mention after each transition, this calculation must be made for the state currently occupied to determine whether to use the optimal or experimental alternative.

It should be noted that this is also a convergent scheme in that eventually all of the variances go to zero, so we always follow the "optimal" rather than the experimental alternative.

6.6 Suggestions for future heuristics

Unfortunately, most of the experimentation with the heuristics discussed in this chapter took place early in the project--before we had a clear idea of just what information we wanted to get from our simulations. From our present point of view, we can suggest several further experiments which should be performed:

1.) An "expected value" evaluation of the first heuristic (follow optimal policy for expected process) must be developed. This will involve applying the heuristic to many random draws from the prior to measure its effectiveness.

2.) We feel that heuristics 2 and 3 are inferior because their parameters are too problem dependent. However, it may be that if a *class* of problems is to be solved, it would pay to get values for the parameters for some members of the class, and use these values for all members of the class. When appropriate parameters are used, convergence is very good, but we are in doubt as to whether that is a prior or posterior fact: did we perhaps just find parameters which happen to give good results given both our prior and the sample points?

6.6.1 A possible heuristic

The investigation on the distribution of the average long-run gain described in the last chapter has given rise to a heuristic which we have not yet tested, but which looks like it might be effective.

Recall that the limitation in following the optimal policy of the expected process was that we considered only immediate rewards, and not the possibility of gaining information. We consider now using a modification of policy iteration to find, instead of only the optimal policy of the expected process, the 10 best or so. We could then use simulation or the predictive results of the last chapter to estimate *both* the *mean* and *variance* of the gain for each of these policies. This would give us measures of both immediate gain and uncertainty, so a trade-off could be established between them. There is much to be learned about a high-variance policy, and much to be gained from a high-mean policy.

It might also be wise to check for *excessively* high variances before entering the above routine, and force tests to eliminate any such.

As is demonstrated in Appendix C, it is impossible to modify the policy iteration method to produce the 10 best policies; but Appendix C continues to present an approximation procedure for finding N of the best policies in an N state process.

There are several experiments which we can suggest as useful for evaluating this heuristic:

1.) By simulation, explore how much actual overlap there is in the range of g for different "good " policies. That is, would a "second best" policy have a good chance of having a large percentage of its values of g being above the $\bar{g}$ of the best policy?

2.) Check by simulation whether for two policies A and B, the relationship between E ($\tilde{g}^A$) and E ($\tilde{g}^B$) is indeed similar to the relationship between g^A and g^B. If this is not the case, knowledge of the "10 best" policies of the expected process does not tell us anything about the "10 best" of the primary process. Our previous experimentation with distribution of gain suggests that the required relationship will hold.

3.) Check, low g and high σ_g^2, whether g is equally likely to increase or decrease. If it should happen to decrease more often than increase (which could be a structural property of this type of process), experimentation with high-variance policies might not be as good an idea as it appears.

On heuristic grounds, this technique seems to be sound. We conclude with a summary of the technique, and a suggestion that it be evaluated:

1.) Find N best policies
2.) Determine mean and variance of the gain for each policy.
3.) Choose a policy by maximizing a weighted sum of mean and variance.
4.) Use that policy for a while; then up-date prior and repeat.

CHAPTER VII

CONCLUSION

The most basic conclusion which emerges from our research and experimentation is that the expected process gives a surprisingly good picture of the primary process. Even for relatively small values of the parameters N_i, the basic statistics of the expected process are good approximations to the expected values of the corresponding statistics for the primary process.

This result lends experimental justification to the most natural approach to the problem. In most current applications, the transition probabilities are not known with certainty anyhow; and some sort of "best estimates" are used. The Bayesian analysis only suggests a formal way of providing these best estimates, and for up-dating them in time.

Still, far more simulation experience is needed before these conclusions can be stated with certainty. It may still be that extreme points from the prior will present enough difficulty so that the expected process approach will not be good in an expected value sense. If further research should indicate that this is the case, then the heuristics dealing with variances as well as means would become more relevant, and would have to be carefully evaluated.

It appears at present that an exact solution to the problem is not feasible. Some sort of approximation procedure will be necessary to handle any good sized problem, and the remaining question is only: which approximation is best?

Finally, there is the matter of relative cost of obtaining solutions. In this regard, it is significant that the expected process technique is so easy to apply. Essentially, it involves only policy iteration, which

Dr. R. A. Howard has shown to be quite practical even for problems with 50 states and 50 alternatives in each state.

Thus, it seems that heuristic methods provide a feasible and useful mean of dealing with Markovian decision processes with uncertain transition probabilities.

APPENDIX A

DETAILS AND LISTINGS OF COMPUTER PROGRAMS

A.1 General Description

Our project called for the testing of many different heuristics, and to facilitate this task, we decided to write our computer programs in blocks, using the subprogram feature of 7090 FORTRAN. Each block was designed as an independent unit, fulfilling a particular task. Then, design of a new heuristic merely required writing a short program to couple these blocks appropriately. Briefly, the subprograms are:

IPUT: Reads in an entire process (prior parameters and rewards).

VALUE(L): A programmed version of R. A. Howard's policy iteration algorithm. Used to find optimal policies and gains.

ITER(L): Used in conjuction with VALUE.

OBS(NOBS): Simulates NOBS transitions of the process and updates the prior accordingly.

ISIM(IPRES): Called by OBS, this subprogram merely simulates a single transition and reports tne outcome.

OPUT(I,N): Causes the printing of results.

PRIOR: Restores the original prior in place of a posterior. Reinitializes between runs.

GEN: Draws a random sample point from the prior distribution.

We now turn to a more detailed account of the subprograms and the MAINS used to tie them together.

A. 2 Subroutine IPUT

The source statement CALL IPUT causes the following cards to be read in:

1.) NS, or number of states in format 12.
2.) NA(I), or the number of alternatives in each state, in format 12, 3X, 12, 3X, etc.
3.) Cards with the prior distribution. Seven entries per card, 10 columns per entry. First the P (I, J), then N_i (all in floating point).
4.) Cards with the rewards. Seven entries per card, 10 columns per entry.

The program stores the values in the proper locations, computes the q_i values, and checks that all probabilities add to one. If an error is found, the message

PROB NOT = 1 IN ROW ____________

prints, and the program terminates. The prior is stored in its working matrix, but also in OPR for reinitialization.

A. 3 Function VALUE (L)

This function is called by a source statement of the form

GAIN = VALUE (L)

If L = 1, it computes the relative values v_i for the *actual* probabilities under the *current* policy.

L = 2, it computes the relative values v_i for the *actual* probabilities under the *optimal* policy.

L = 3, it computes the relative values v_i for the prior probabilities under the *current* policy.

L = 4, it computes the relative values v_i for the prior probabilities under the optimal policy.

L = 5, computes the q_i values for both actual and prior probabilities.

GAIN, which is the long run expected gain associated with the computed values v_i, is returned as the functional value. If an optimal policy is computed, it is left in the K(I) vector of common storage. The values are always left in the V(I) vector of common storage.

Thus, for example, the sta‘ement GAIN = VALUE(4) would cause the optimal policy of the expected process, along with its values and gain, to be computed.

A.4 Function ITER (L)

This subroutine is called by the VALUE subprogram, and corresponds to the "policy improvement" phase of R. A. Howard's algorithm.

If L = 1, it improves the policy assuming actual probabilities.

L = 2, same as L = 1.

L = 3, it improves the policy assuming prior probabilities

L = 4, same as L = 3.

L = 5, it chooses an initial policy for the actual probabilities by maximizing q_i in each state.

L = 6, it chooses an initial policy for the prior probabilities by maximizing q_i in each state.

It is called by a source statement of the form

IOPT = ITER (L),

and IOPT is returned as 1 if the policy did not change (optimum found), and as 2 if the policy did change (optimum not yet found).

A. 5 Subroutine OBS (NOBS)

This subprogram is called by a source statement of the form,

```
CALL OBS ( NOBS )
```

It simply causes NOBS transitions to be simulated using the actual probabilities, and the frequencies to be tallied. After the observations are completed, the subroutine performs a Bayesian up-dating of the prior before returning control to the main program.

A. 6 Function ISIM (IPRES)

This function is called by the OBS subprogram, and it is this subroutine which actually does the simulation of transitions. A random number is drawn from a retangular (0, 1) distribution, and this random number in conjuction with the transition probabilities determines a transition. The new state is reported back to CBS, and the latter program records it, etc.

A. 7 Subroutine PRIOR

This subroutine simply reinitializes the prior matrix, and calls GEN to supply a new sample process. PRIOR is the recycle point of all the MAINS.

A. 8 Subroutine OPUT (I, N)

This subroutine performs several different functions, since many types of output are needed. Ostensibly, I is the iteration number, and N is the number of observations in the iteration. If both I and N are non-zero, normal output results. Normal output consists of the following:

1.) The iteration number I

2.) The number of observations N

3.) The estimated gain, or gain of process using prior probabilities and current policy. The program assumes that this quantity has been computed and is in the common storage location GAIN.

4.) The actual gain of the current policy. This is computed by the OPUT subroutine from the actual probabilities before printing.

5.) Accumulated number of observations. A quantity computed by OPUT.

6.) Efficiency. This too is computed by OPUT, and is defined by:

$$\frac{\text{Previous accumulated profit} + (\text{no. obs.}) \times (\text{actual gain})}{(\text{Optimal actual gain of process}) \times (\text{accumulated no. of obs.})},$$

where the numerator becomes the new previous accumulated profit. The optimum actual gain of the process is known by OPUT (see below).

7.) The current policy being followed. This must be in the K(I) vector when OPUT is called.

The source statement CALL OPUT (I, N) causes a single line with this information to be printed, if both I and N are positive. But, I can also serve two control functions:

<u>I = 0:</u>

This causes the process information to be printed. The prior, reward, and actual probabilities matrices are all printed, and control is immediately returned to the main program.

$I<0$

This causes initialization preparatory to a new run. The actual optimum gain is computed and stored for later use in computing efficiency. Then a header is printed to identify later output. Finally, a line identified as iteration 0 is printed in which: estimated gain = actual gain = optimum actual gain of the process. The policy indicated is the optimal policy, and all other quantities are zero. The output from this section looks like:

```
RUN NO.

       ____________

IT   NO OBS   EST GAIN   ACT GAIN   ACCUM OBS   EFF   POLICY

0      0      (actual optimum gain)      0        0  (actual optimum
                                                          policy)
```

There is a special OPUT program for MAIN 4 which gives slightly different output, and computes EFF on the basis of actually observed rewards.

A.9 Subroutine GEN

This subroutine is called by a source statement of the form

```
CALL GEN
```

The statement causes a sample process to be drawn from the prior and placed in the "actual probabilities" locations. The q_i values are computed by GEN, and control is returned to the main program.

The sampling is done according to the method outlined in section 2.4. As mentioned there, trucation is used to assure that all parameters are integral, or else simulation would not be possible.

A.10 MAIN I

This is a programmed version of the first heuristic described in Chapter 6: follow optimal policy for expected process. It needs a parameter card with the following items:

1.) IPRES, present state of system (initial), in columns 1-2.

2.) NOBS, number of observations between recomputations of policy, in columns 6-8.

3.) NOITS, number of iterations for each sample process, in columns 12-14.

4.) IRUN, run number, in columns 18-19.

The process information should follow this card (see IPUT).

A process is generated and simulation for NOITS iterations of NOBS observations each. Then a new process is generated, and so forth. There is no provision for termination of the program, we just run until running out of time.

A flow chart follows.

A.11 MAIN 2

This is a programmed version of the second heuristic described in Chapter 6, a trade-off between immediate rewards and information. The following parameter card is expected:

1.) IPRES, initial state of system, in columns 1-2.

2.) IRUN, the run number, in columns 6-7.

3.) NOITS, the desired number of iterations, in columns 11-12.

4.) ALPHA, (see Section 6.3) in columns 16-20.

5.) BETA, (see Section 6.3) in columns 24-28.

The process information follows this card (see IPUT).

A process is generated, and simulation for NOITS iterations. Then a new process is generated, etc. No termination is provided for--program runs until stopped.

A flow chart follows.

A.12 MAIN 3

This is a programmed version of the third heuristic described in Chapter 6, weighted immediate reward technique. The parameter card contains:

1.) IRUN, the run number, in columns 1-2.

2.) IPRES, the initial state of the system, in columns 6-7.

3.) NOITS, the desired number of iterations, in columns 11-12.

4.) ALPHA, (see Section 6.4) in columns 16-20.

5.) BETA. (see Section 6.4) in columns 24-28.

The process should follow (see IPUT).

A process is generated and simulation for NOITS iterations occurs. Then a new sample process is generated, etc. No termination is provided for.

A flow chart follows.

A. 13 MAIN 4

This is a programmed version of the fourth heuristic described in Chapter 6: the weighted variance technique. The parameter card contains:

1.) IRUN, the run number, in columns 1-3.

2.) IPRES, the initial state of the system, in columns 7-9.

3.) NOITS, the desired number of iterations, in columns 13-15.

4.) CEE (see Section 6.5), in columns 19-28.

The process information follows the parameter card (see IPUT).

A sample process is generated and simulation for NOITS iterations occurs. Then a new sample process is generated, etc. No termination is provided for.

A flow chart follows.

A. 14 Distribution of gain

This is a programmed version of the algorithm discussed in Chapter 5. The first data card must be a parameter card:

1.) IRUN, run number, in columns 1-3.

2.) MAX, or maximum number of sample points, in columns 7-10.

3.) MTIM, in columns 14-17.

MTIM is used in conjuction with the MIT clock-interrupt, and specifies the latest time at which sampling should end and output begin, measured from the termination time of the program. If MTIM is set at less than 200, it will insure that no output is lost by running overtime, while it will not interfere with normal running if overtime would not occur.

The program reads only one parameter card, but after each output it returns to read the information for another process. So, by stacking the information for several processes behind the parameter card, any number can be investigated.

NOTE: Since gain refers to a policy, the process supplied must have only one alternative per state.

A flow chart follows.

A. 15 Solution of the special two-state problem

This is a programmed version of the solution routine described for a two-state process in Chapter 3. It requires three data cards.

1.) BETA and NMAX, the value to be taken as infinity.

2.) The known probabilities P (1, 1), P(1, 2), P(2, 1), P(2, 2).

3.) The rewards for the known and unknown alternatives:

R(1, 1), R(1, 2), R(2, 1), R(2, 2), R(3, 1), R(3, 2)

All entries are 10 columns each, floating point, except NMAX which is a 5 position integer.

The program prints a column of values for each $n < 50$, $0 \leq r \leq n$. When all columns have been printed, a summary chart of the decision boundary is printed.

A.16 PROGRAM LISTINGS

```
      SUBROUTINE IPUT
C     SUBROUTINE FOR DATA INPUT
C
      DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE  (A,W),(W(226),WORK2)
      READ 12, NS
      READ 12, (NA(I), I=1,NS)
      NUM = NA(1)
      NA(1) = 0
      DO 5  I=1,NS
      NX = NA(I+1)
      NA(I+1) = NUM
    5 NUM = NUM + NX
      NL = NS + 1
      N = NA(NL)
      DO 6  I =1,N
    6 READ 11, (P(I,J), J=1,NL)
      DO 7  I=1,N
    7 READ 11, (R(I,J), J=1,NS)
      DO 8 I=1,N
      DO 8 J=1,NL
    8 OPR(I,J)=P(I,J)
      DUM = VALUE(5)
   11 FORMAT (7F10.6)
   12 FORMAT (I2,14(3X,I2))
      DO 15  I=1,N
      SUM = 0.0
      SUM2 = 0.0
      DO 14  J=1,NS
   14 SUM = SUM + P(I,J)
      ERF = .0001
      IF (ABSF(1.-SUM)-ERF) 15,18,18
   15 CONTINUE
      RETURN
   18 PRINT 19, I
   19 FORMAT (21H1PROB NOT = 1 IN ROW ,I2)
      CALL EXIT
      END
      FUNCTION  VALUE(L)
C     VALUE SUBPROGRAM   10-16-63
      DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE  (A,W),(W(226),WORK2)
      IOPT = 2
      NSM=NS-1
    7 GO TO (1,1,2,2,3),L
    1 DO 4 I=1,NS
      LL=NA(I)+K(I)
      WORK2(I)=AP(LL,NS+1)
    5 DO 4 J=1,NSM
    4 A(I,J) = -AP(LL,J)
    6 DO 8 I=1,NS
      A(I,I) = A(I,I) + 1.
    8 A(I,NS) = 1.
C     CALL LINEAR EQUATION SOLUTION ROUTINE
      SCALE = 1.
      M = XSIMEQF(15,NS,1,A,WORK2,SCALE,V)
    9 GO TO (10,11,11),M
   10 DO 12 I=1,NSM
   12 V(I) = A(I,1)
      V(NS) = 0.
      VALUE = A(NS,1)
   13 GO TO (14,15,14,15),L
C     CALL ITERATION ROUTINE
   15 IOPT = ITER(L)
   16 GO TO (14,7),IOPT
C     IF L = 3 OR 4, GO HERE
    2 DO 17 I=1,NS
      LL=NA(I)+K(I)
      WORK2(I) = P(LL,NS+2)
   18 DO 17 J=1,NSM
   17 A(I,J) = -P(LL,J)
      GO TO 6
C     IF L=5,COME HERE
    3 NSM = NA(NS+1)
   19 DO 20 I=1,NSM
      AP(I,NS+1) =0.
      P(I,NS+2) = 0.
   21 DO 20 J=1,NS
      AP(I,NS+1) = AP(I,NS+1) + AP(I,J)*R(I,J)
   20 P(I,NS+2) = P(I,NS+2) + P(I,J)*R(I,J)
      VALUE = 0.
      FREQUENCY 7(1,1,1,7,1),1(15),5(15),6(15),9(1,0,0),10(15)
      FREQUENCY 13(1,1,1,7),16(1,5),2(15),18(15),19(15),21(15)
   14 RETURN
C     ERROR EXIT
   11 PRINT 30
   30 FORMAT (23H NO SOLUTION FOR VALUES)
      CALL EXIT
      END
      FUNCTION  ITER(L)
C     ITERATION SUBPROGRAM   10-16-63
C
      DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE  (A,W),(W(226),WORK2)
      ITER = 1
    2 DO 1 I=1,NS
      TEMP = -99999.
      NTEMP = 0
      IMIN = NA(I) +1
      IMAX = NA(I+1)
    3 DO 15  M=IMIN,IMAX
    5 GO TO (6,6,7,7,8,9),L
    6 TEST = AP(M,NS+1)
   10 DO 11  J=1,NS
   11 TEST = TEST + AP(M,J)*V(J)
      GO TO 12
    7 TEST = P(M,NS+2)
   14 DO 13 J=1,NS
   13 TEST = TEST + P(M,J)*V(J)
      GO TO 12
    8 TEST = AP(M,NS+1)
      GO TO 12
    9 TEST = P(M,NS+2)
   12 IF (TEST-TEMP) 15,16,17
   16 IF (NTEMP - K(I)) 17,15,17
   17 NTEMP = M-NA(I)
      TEMP = TEST
   15 CONTINUE
   18 IF (NTEMP - K(I)) 19,1,19
   19 ITER = 2
      K(I) = NTEMP
    1 CONTINUE
      RETURN
      FREQUENCY  2(15),3(13),5(1,1,60,60,1,20),10(15),14(15),12(10,0,10)
      FREQUENCY 16(1,1,1),18(1,2,1)
      END
      SUBROUTINE OBS(NOBS)
C     SUBROUTINE FOR OBSERVING THE PROCESS AND FOR
C        UPDATING THE ESTIMATED PROBABILITIES AND THE Q'S
C
      DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE  (A,W),(W(226),WORK2)
      NS1 = NS + 1
      DO 2  I=1,NS
      DO 2  J=1,NS1
    2 W(I,J) = 0.0
      DO 3  I=1,NOBS
      INEXT = ISIM(IPRES)
      W(IPRES,INEXT) = W(IPRES,INEXT) + 1.0
      W(IPRES,NS1) = W(IPRES,NS1) + 1.0
    3 IPRES = INEXT
      DO 6  I=1,NS
      IF (W(I,NS1))  1,6,4
C     COMPUTE NEW Q(I)
    4 IP = K(I) + NA(I)
      OLDEN = P(IP,NS1)
      P(IP,NS+1) = OLDEN + W(I,NS1)
      P(IP,NS+2) = 0.0
      DO 5 J=1,NS
      P(IP,J) =(P(IP,J)*OLDEN + W(I,J))/P(IP,NS+1)
    5 P(IP,NS+2) = P(IP,NS+2) + P(IP,J)*R(IP,J)
    6 CONTINUE
      RETURN
    1 PRINT 501
  501 FORMAT (10H ERROR 501)
      CALL EXIT
      END
      FUNCTION ISIM(IPRES)
C
C     SIMULATE ONE OBSERVATION
C
      DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE  (A,W),(W(226),WORK2)
      IP = K(IPRES) + NA(IPRES)
      AA = RANNOF(X)
      DO 2  J=1,NS
      AA = AA - AP(IP,J)
      IF (AA)  3,3,2
    3 ISIM = J
      GO TO 5
    2  CONTINUE
      ISIM = NS
    5 RETURN
      END
      SUBROUTINE OPUT(I,N)
C     OPUT 3  ONE LINE OUTPUT
C     I IS THE ITERATION NUMBER
C     N IS THE NUMBER OF OBSERVATIONS IN THE ITERATION
      DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE  (A,W),(W(226),WORK2)
      DIMENSION HOLD(15)
C
      IF (I)  52,3,2
C     IF I=0, PRINT MATRICES
    3 PRINT 501,IRUN
  501 FORMAT (8H1RUN NO ,I4)
C     PRINT ACTUAL PROBABILITY MATRIX
      PRINT 505
  505 FORMAT (26H1ACTUAL PROBABILITY MATRIX)
      LA=1
   13 PRINT 506,(L,L=1,10)
  506 FORMAT (6X,10(6X,I2),5X,3HOBS,11X,1HQ)
      DO 9 L=1,NS
      MAX = NA(L+1)-NA(L)
      DO 9 KA=1,MAX
      IP = NA(L)+KA
      GO TO (6,7,8),LA
    6 PRINT 507,L,KA,(A (IP,J),J=1,NS   )
  507 FORMAT (1H0,I2,1H,I2,2X,10F8.4/(8X,10F8.4))
      PRINT 531, AP(IP,NS+1)
  531 FORMAT (1H+,99X,F10.4)
      GO TO 9
    7 PRINT 513,L,KA,(R(IP,J),J=1,NS)
  513 FORMAT (1H0,I2,1H,I2,2X,10F8.2/(8X,10F8.2))
      GO TO 9
    8 PRINT 507,L,KA,(P(IP,J),J=1,NS    )
      PRINT 520, P(IP,NS+1),P(IP,NS+2)
  520 FORMAT (1H+,88X,F10.0,1X,F10.4)
    9 CONTINUE
      GO TO (10,11,12) , LA
C     PRINT REWARD MATRIX
   10 LA = 2
      PRINT 508
  508 FORMAT (14H1REWARD MATRIX)
      GO TO 13
   11 LA = 3
      PRINT 514
  514 FORMAT (29H1ESTIMATED PROBABILITY MATRIX)
      GO TO 13
C     COMPUTE TRUE OPTIMUM POLICY.
```

```
   12 RETURN
   52 MAX = ITER(5)
      OGAIN = VALUE(2)
      PRINT HEADER
      PRINT 530, IRUN
  530 FORMAT (8H1RUN NO ,I4/51H  IT  NO OBS   EST GAIN   ACT GAIN ACCUM
     1OBS    EFF,5X,6HPOLICY)
      EGAIN = OGAIN
      AGAIN = OGAIN
      ACCOBS = 0.0
      PROF = 0.0
      EFF = 0.0
      I = 0
   50 PRINT 502,I,N,EGAIN,AGAIN,ACCOBS,EFF,(K(J),J=1,NS)
  502 FORMAT (1X,I3,4X,I4,1X,F10.2,1X,F10.2,1X,F9.0,2X,F6.4,2X,I2,14(
     11H,I2))
      RETURN
C     IF NOT I=0,COME HERE
C     STORE CURRENT VALUES FO COMPUTE ACTUAL VALUES.
    2 DO 14 KA=1,NS
   14 HOLD(KA) = V(KA)
      AGAIN = VALUE(1)
      EGAIN = GAIN
      DO 16 KA = 1,NS
   16 V(KA) = HOLD(KA)
      FNOBS = N
      PROF = PROF + FNOBS*AGAIN
      ACCOBS = ACCOBS+FNOBS
      EFF = PROF/(OGAIN*ACCOBS)
      GO TO 50
      END
      SUBROUTINE OPUT(I,N)
C     OPUT 4   FOR USE WITH MAIN 4 ONLY -- EFF IN TERMS OF REWARDS
C     I IS THE ITERATION NUMBER
C     N IS THE NUMBER OF TIMESS OPTIMUM WAS USED
      DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE  (A,W),(W(226),WORK2)
      DIMENSION HOLD(15)
C
      IF(I) 2,3,2
C     IF I=0, PRINT MATRICES
    3 PRINT 501,IRUN
  501 FORMAT (8H1RUN NO ,I4)
C     PRINT ACTUAL PROBABILITY MATRIX
      PRINT 505
  505 FORMAT (26H1ACTUAL PROBABILITY MATRIX)
      LA=1
   13 PRINT 506,(L,L=1,10)
  506 FORMAT (6X,10(6X,I2),5X,3HOBS,11X,1HQ)
      DO 9 L=1,NS
      MAX = NA(L+1)-NA(L)
      DO 9 KA=1,MAX
      IP = NA(L)+KA
      GO TO (6,7,8),LA
    6 PRINT 507,L,KA,(AP(IP,J),J=1,NS   )
  507 FORMAT (1H0,I2,1H,I2,2X,10F8.4/(8X,10F8.4))
      PRINT 531, AP(IP,NS+1)
  531 FORMAT (1H+,99X,F10.4)
      GO TO 9
    7 PRINT 513,L,KA,(R(IP,J),J=1,NS)
  513 FORMAT (1H0,I2,1H,I2,2X,10F8.2/(8X,10F8.2))
      GO TO 9
    8 PRINT 507,L,KA,(P(IP,J),J=1,NS    )
      PRINT 520, P(IP,NS+1),P(IP,NS+2)
  520 FORMAT (1H+,88X,F10.0,1X,F10.4)
    9 CONTINUE
      GO TO (10,11,12) , LA
C     PRINT REWARD MATRIX
   10 LA = 2
      PRINT 508
  508 FORMAT (14H1REWARD MATRIX)
      GO TO 13
   11 LA = 3
      PRINT 514
  514 FORMAT (29H1ESTIMATED PROBABILITY MATRIX)
      GO TO 13
C     COMPUTE TRUE OPTIMUM POLICY.
```

```
   12 MAX = ITER(5)
      OGAIN = VALUE(2)
C     PRINT HEADER
      PRINT 530, IRUN
  530 FORMAT (8H1RUN NO ,I4/51H  IT  NO OPT   EST GAIN   ACT GAIN ACCUM
     1OBS    EFF,5X,6HPOLICY)
      EGAIN = OGAIN
      AGAIN = OGAIN
      ACCOBS = 0.0
      PROF = 0.0
      EFF = 0.0
   50 PRINT 502,I,N,EGAIN,AGAIN,ACCOBS,EFF,(K(J),J=1,NS)
  502 FORMAT (1X,I3,4X,I4,1X,F10.2,1X,F10.2,1X,F9.0,1X,F6.2,2X,I2,14(
     11H,I2))
      RETURN
C     IF NOT I=0,COME HERE
C     STORE CURRENT VALUES FO COMPUTE ACTUAL VALUES.
    2 PROFIT = WORK2(1)
      AGAIN = VALUE(1)
      PROF = PROF + PROFIT
      ACCOBS = ACCOBS + 100.
      EFF = (PROF/(OGAIN*ACCOBS))*100.
      EGAIN = GAIN
      GO TO 50
      END
      SUBROUTINE PRIOR
C     RESTORES ORIGINAL PRIOR AND GENERATES A SAM  E PROCESS
C
      DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE  (A,W),(W(226),WORK2)
      K=NS+2
      MAX = NA(K-1)
      DO 1 I=1,MAX
      DO 1 J=1,K
    1 P(I,J)=OPR(I,J)
      CALL GEN
      RETURN
      END
      SUBROUTINE GEN
C     GENERATOES A SAMPLE PROCESS
C     SPECIFICATIONS
      DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE  (A,W),(W(226),WORK2)
      N = NA(NS+1)
      DO 22 I=1,N
      SUM = 0.
      DO 20  J=1,NS
      M = P(I,J)*P(I,NS+1)
      RAN =  0.
      DO 21  K=1,M
      AA = RANNOF(X)
   21 RAN = RAN - .05*LOGF(AA)
      AP(I,J) = RAN
   20 SUM = SUM + AP(I,J)
      DO 22  J=1,NS
   22 AP(I,J) = AP(I,J)/SUM
      DUMMY = VALUE(5)
      RETURN
      END
```

```
C       MAIN 1    5/5/64
C       FOLLOW OPTIMAL POLICY OF EXPECTED PROCESS
C
        DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
        DIMENSION V(15),WORK2(15),A(15,15)
        DIMENSION OPR(150,17)
        COMMON OPR
        COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
        EQUIVALENCE  (A,W),(W(226),WORK2)
        READ 502, IPRES,NOBS,NOITS,IRUN
        CALL IPUT
    502 FORMAT (I2,3X,I3,3X,I3,3X,I2)
        CALL OPUT(0,0)
C       CHOOSE AN ITITIAL POLICY USING ESTIMATES
        IDUM = ITER(6)
C       WASTE 25 RANDOM NUMBERS
        AA = SETUF(IRUN)
        DO 7 I=1,25
      7 AA=RANNOF(X)
      5 CALL PRIOR
        CALL OPUT (-1,0)
C       SIMULATE NOITS ITERATIONS
        DO 2 I=1,NOITS
        GAIN = VALUE(4)
        CALL OBS(NOBS)
        CALL OPUT(I,NOBS)
      2 CONTINUE
        GO TO 5
        END
```

```
C     MAIN 2              5/15/64
C     TRADE OFF BETWEEN IMMEDIATE GAIN AND INFORMATION
      DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE  (A,W),(W(226),WORK2)
      DIMENSION  HOLD(150)
      READ 501,IPRES,IRUN,NOITS,ALPHA, BETA
  501 FORMAT (I2,3X,I2,3X,I2,3X,F5.2,3X,F5.2)
      CALL IPUT
C     WASTE 25 RANDOM NUMBERS
      AA=SETUF(IRUN)
      DO 7 I=1,25
    7 AA = RANNOF(X)
      NS = NS
      MAX = NA(NS+1)
      CALL OPUT(0,0)
    6 CALL PRIOR
      CALL OPUT (-1,0)
      DO 1 KKK=1,NOITS
C     PUT Q'S IN HOLDING AREA AND RELATIVIZE
      QMIN = 999999.
      DO 2 I=1,MAX
      TEMP = P(I,NS+2)
    2 QMIN = MIN1F(QMIN,TEMP)
      DO 3 I=1,MAX
      HOLD(I) = P(I,NS+2)-QMIN
    3 P(I,NS+2) = (HOLD(I)**ALPHA)/P(I,NS+1)
C     COMPUTE POLICY BY MAXIMIZING W
      IDUM = ITER(6)
C     FIND MINIMUM W
      WMIN = 999999.
      DO 4 I=1,NS
      IDUM = K(I)+NA(I)
      TEMP  = P(IDUM,NS+2)
    4 WMIN = MIN1F(WMIN,TEMP)
C     RESTORE Q'S
      DO 5 I=1,MAX
    5 P(I,NS+2) = HOLD(I)+QMIN
C     OBSERVE PROPORTIONAL TO WMIN
      NOBS = (BETA*WMIN) + 1.
      GAIN = VALUE(3)
      CALL OBS(NOBS)
      CALL OPUT(KKK,NOBS)
    1 CONTINUE
      GO TO 6
      END
```

```
C     MAIN 3
C      WEIGHTED REWARDS RECHNIQUE
C     C=(ALPHA*Q)/N +1
      DIMENSIONNA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE (A,W),(W(226),WORK2)
      DIMENSION C(150)
      READ 501,IRUN,IPRES,NOITS,ALPHA,BETA
  501 FORMAT (3(I2,3X),F5.0,3X,F5.0)
      CALL IPUT
      CALL OPUT (0,0)
C     WASTE 25 RANDOM NUMBERWS
      AA = SETUF(IRUN)
      DO 2 I=1,25
    2 AA = RANNOF(X)
      IDUM=ITER(6)
      NS=NS
      MAX = NA(NS+1)
   99 CALL PRIOR
      CALL OPUT(-1,0)
      DO 3 I=1,NOITS
      DO 4 J=1,MAX
      C(J) = (ALPHA*ABSF(P(J,NS+2)))/P(J,NS+1) + 1.0
    4 P(J,NS+2) = P(J,NS+2)*C(J)
      GAIN = VALUE(4)
      FNOBS = 0.
      DO 7 J=1,MAX
    7 P(J,NS+2) = P(J,NS+2)/C(J)
C     COMPUTE EST. GAIN FOR POLICY USING NON-DUMMY Q'S
      GAIN = VALUE(3)
      DO 5 J=1,NS
      IMIN = NA(J) +1
      IMAX = NA(J+1)
      CMIN = 999999.
      DO 6 L=IMIN,IMAX
      IF (CMIN-C(L)) 6,6,8
    8 CMIN = C(L)
      ICMIN = L
    6 CONTINUE
      IDUM = NA(J) + K(J)
      IF(IDUM-ICMIN) 10,11,10
   11 FNOBS = FNOBS + BETA*P(IDUM,NS+1)
      GO TO 5
   10 FNOBS=(P(ICMIN,NS+1)*ABSF(P(IDUM,NS+2)))/ABSF(P(ICMIN,NS+2))-
     1P(IDUM,NS+1) + FNOBS
    5 CONTINUE
      NOBS = FNOBS
      CALL OBS(NOBS)
      CALL OPUT(I,NOBS)
    3 CONTINUE
      GO TO 99
      END
```

```
C     MAIN 4  WEIGHTED VARIANCE TECHNIQUE
      DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE  (A,W),(W(226),WORK2)
      DIMENSION Q(20)
      READ 501, IRUN,IPRES,NOITS,CEE
  501 FORMAT (I3,3X,I3,3X,I3,3X,F10.5)
C     WASTE SOME RANDOM NUMBERS
      AA = SETUF(IRUN)
      DO 2 I = 1,25
    2 AA = RANNOF(X)
      CALL IPUT
   14 CALL PRIOR
      CALL OPUT (0,0)
      DO 3 JJJ=1,NOITS
      NOPTS = 0
      PROFIT = 0.0
      DO 4 KKK = 1,100
      EGAIN = VALUE(4)
      KOPT = K(IPRFS)
      MIN = NA(IPRES)+1
      MAX = NA(IPRES+1)
      VAR = 0.0
      NALT = 0
      DO 5 I=MIN,MAX
      TEST = 0.0
      DO 6 J=1,NS
    6 TEST = TEST + R(I,J)*P(I,J)*(1.-P(I,J))
      TEST =(TEST/(((P(I,NS+1))*(P(I,NS+1)+1.))**2))*CEE
      IF (TEST-VAR) 5,5,7
    7 VAR = TEST
      NALT = I
    5 CONTINUE
      IF (NA(IPRFS)+K(IPRES)-NALT) 8,13,8
    8 DO 10 I=MIN,MAX
      J=I-MIN+1
      Q(J) =P(I,NS+2)
   10 P(I,NS+2) = -999999.
      J = NALT - MIN + 1
      P(NALT,NS+2) = Q(J)
      TGAIN = VALUE(4)
      DO 11 I=MIN,MAX
      J=I-MIN+1
   11 P(I,NS+2) = Q(J)
      IF (EGAIN - TGAIN - VAR) 12,12,13
   13 K(IPRES) = KOPT
      NOPTS = NOPTS + 1
```

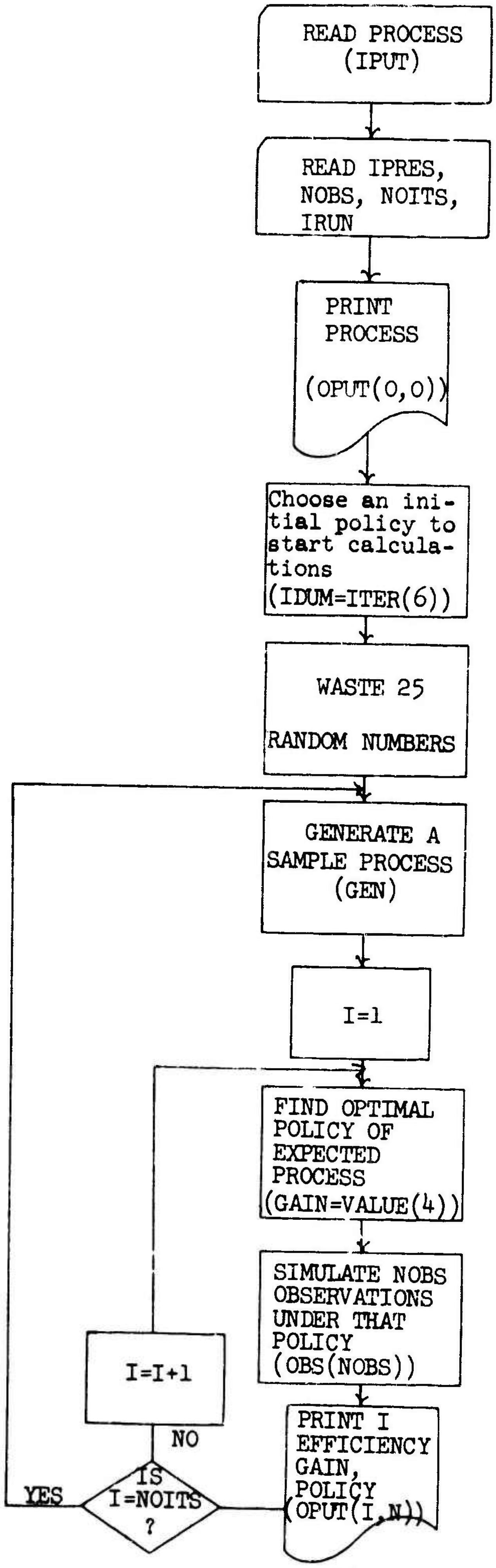
```
 12 IOLD = NA(IPRES) + K(IPRES)
    CALL OBS(1)
    IPRES = IPRES
    PROFIT = PROFIT + R(IOLD,IPRES)
  4 CONTINUE
    GAIN = VALUE(4)
    WORK2(1) = PROFIT
    CALL OPUT (JJJ,NOPTS)
  3 CONTINUE
    GO TO 14
    END
```

```
C     MAIN5   PLOT DISTR. OF GAIN
C     SPECIFICATIONS
      DIMENSION NA(16),K(15),AP(150,16),R(150,16),P(150,17),W(15,16)
      DIMENSION V(15),WORK2(15),A(15,15)
      DIMENSION OPR(150,17)
      COMMON OPR
      COMMON  NA,K,AP,R,P,W,V,NS,IPRES,GAIN,IRUN
      EQUIVALENCE  (A,W),(W(226),WORK2)
      DIMENSION G(51)
      READ 501,IRUN,MAX,MTIM
  501 FORMAT (I3,3X,I4,3X,I4)
C     WASTE SOME RANXOM NUMBERS
      AA = SETUF(IRUN)
      DO 12 I=1,25
   12 AA = RANNOF(X)
   14 CALL IPUT
C     COMPUTE RANGE OF GAIN
      PGAIN = VALUE(4)
      PRINT 503,PGAIN
      RMIN = PGAIN*.85
      RMAX = PGAIN*1.15
      RINC = (RMAX-RMIN)/50.
C     INITIALIZE
      GSUM = 0.
      GSQ = 0.
      DO 3 I=1,51
    3 G(I) = 0.
C     BEGIN SIMULATION LOOP
      DO 6 KKK = 1,MAX
      CALL GEN
      GAIN = VALUE(2)
      GSUM = GSUM + GAIN
      GSQ = GSQ + GAIN*GAIN
      GAIN = GAIN - RINC
      DO 7 JK=1,50
      IF(GAIN-RMIN) 5,5,4
    5 G(JK) = G(JK) +1.
      GO TO 9
    4 GAIN = GAIN - RINC
    7 CONTINUE
      G(51) = G(51) + 1.
    9 FMAX = KKK
      CALL TIMLFT(JTIM)
      IF(JTIM-MTIM) 10,10,6
    6 CONTINUE
   10 GMEAN = GSUM/FMAX
      GVAR = GSQ/FMAX - GMEAN*GMEAN
      PRINT 502,GMEAN,GVAR,PGAIN,FMAX
  502 FORMAT (1H1,6X,4HGAIN,2X,4HPROB,2X,6HMEAN =,F10.4,2X,5HVAR =,F10.4
     1,2X,10HEXP GAIN =,F10.4,2X,10HNO SAMPS =,F6.0)
      DO 8 I=1,51
      G(I) = G(I)/FMAX
      RMIN = RMIN + RINC
      PRINT 503,RMIN,G(I)
  503 FORMAT (1X,F10.2,2X,F10.5)
    8 CONTINUE
      GO TO 14
      END
```

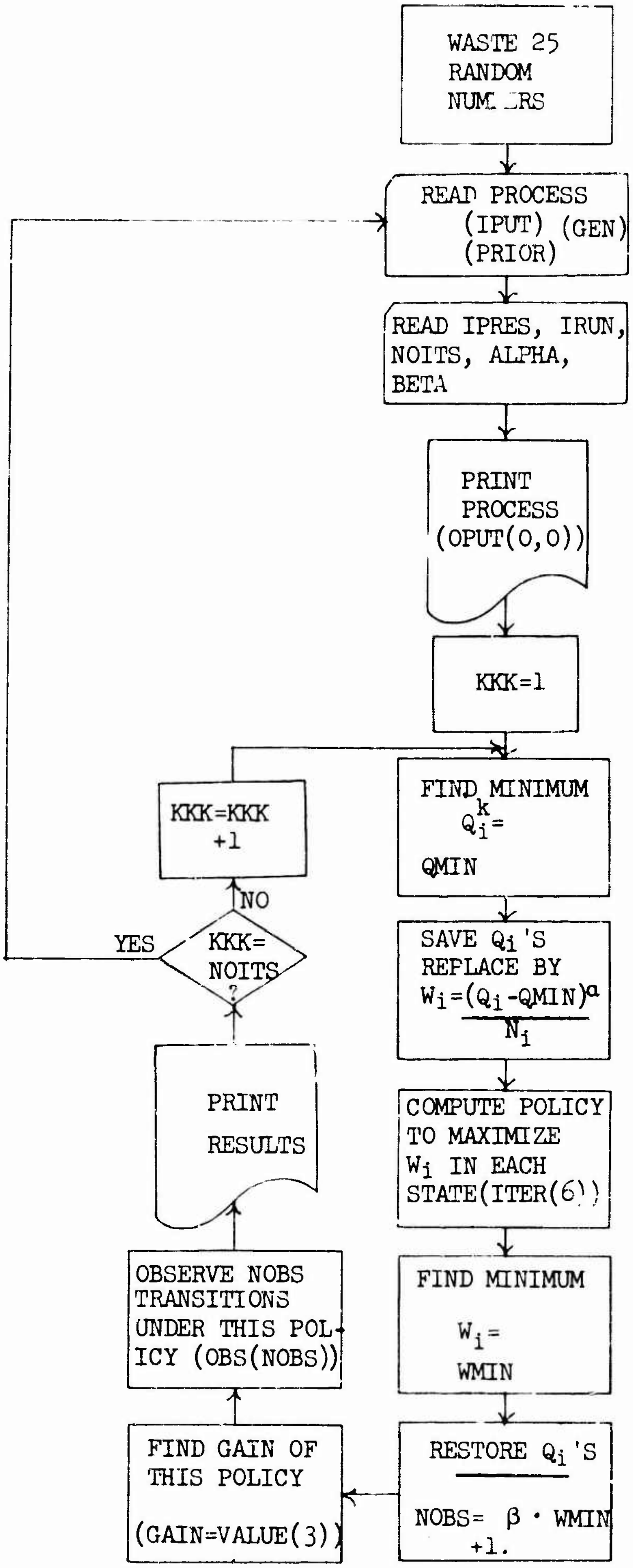
```
C     NUMERICAL SOLUTION TO SPECIAL TWO STATE PROBLEM
      DIMENSION P(2,2), R(3,2), V(1000), IBDY(1000)
      DO 21 II=1,5
    2 READ 101 , BETA,NMAX
      READ 102, P(1,1), P(1,2), P(2,1), P(2,2)
      READ  102, R(1,1), R(1,2), R(2,1), R(2,2), R(3,1), R(3,2)
  101 FORMAT (F10.0,I5)
  102 FORMAT (6F10.0)
      A1 = R(3,1) + BETA*(P(1,1)*R(1,1) + P(1,2)*R(1,2))/(1.0 -
     1  P(1,1)*BETA)
      A2 = (P(1,2)*BETA*BETA)/(1.0 - P(1,1)*BETA )
      A3 = R(3,2)
      A4 = BETA
      C = (A4*P(2,1)*(P(1,1)*R(1,1) + P(1,2)*R(1,2)) + (1.0 - A4*P(1,1))
     1  *(P(2,1)*R(2,1) + P(2,2)*R(2,2)))/((1.0 - A4*P(1,1))*(1.0 - A4*
     2   P(2,2)) - P(1,2)*P(2,1)*A4*A4)
      DO 5 I=1,NMAX
      FN=NMAX
      S=I
      V(I) = (S*A1/FN + (1.0 - S/FN)*A3)/(1.0 - S*A2/FN - (1.0 - S/FN)*
     1   A4)
      IF(V(I)-C)6,6,5
    6 V(I)=C
      MAXR=I
      IBDY(NMAX)=MAXR
      GO TO 7
    5 CONTINUE
      GO TO 3
    7 DO 4 I=MAXR,NMAX
    4 V(I)=C
    3 NM=NMAX-1
   20 FORMAT (I5,I5/(10F10.4))
      DO 12 NN=1,NM
      N=NMAX-NN
      FN=N
      DO 8 J=1,N
      S=J
    9 V(J) = (S*(A1+A2*V(J+1))/FN)+(1.-S/FN)*(A3+A4*V(J))
      IF(V(J)-C)10,10,8
   10 V(J) = C
      IBDY(N)=J
      GO TO 11
    8 CONTINUE
   11 IBDN=IBDY(N)
      IF(N-51)300,300,301
  300 PRINT 20, N,IBDN,(V(I),I=1,IBDN)
  301 CONTINUE
   12 CONTINUE
      PRINT 105, (IBDY(J),J=1,NMAX)
  105 FORMAT (1X,20I3,2X/)
   21 CONTINUE
      CALL EXIT
      END
```

A. 17 Flow Charts

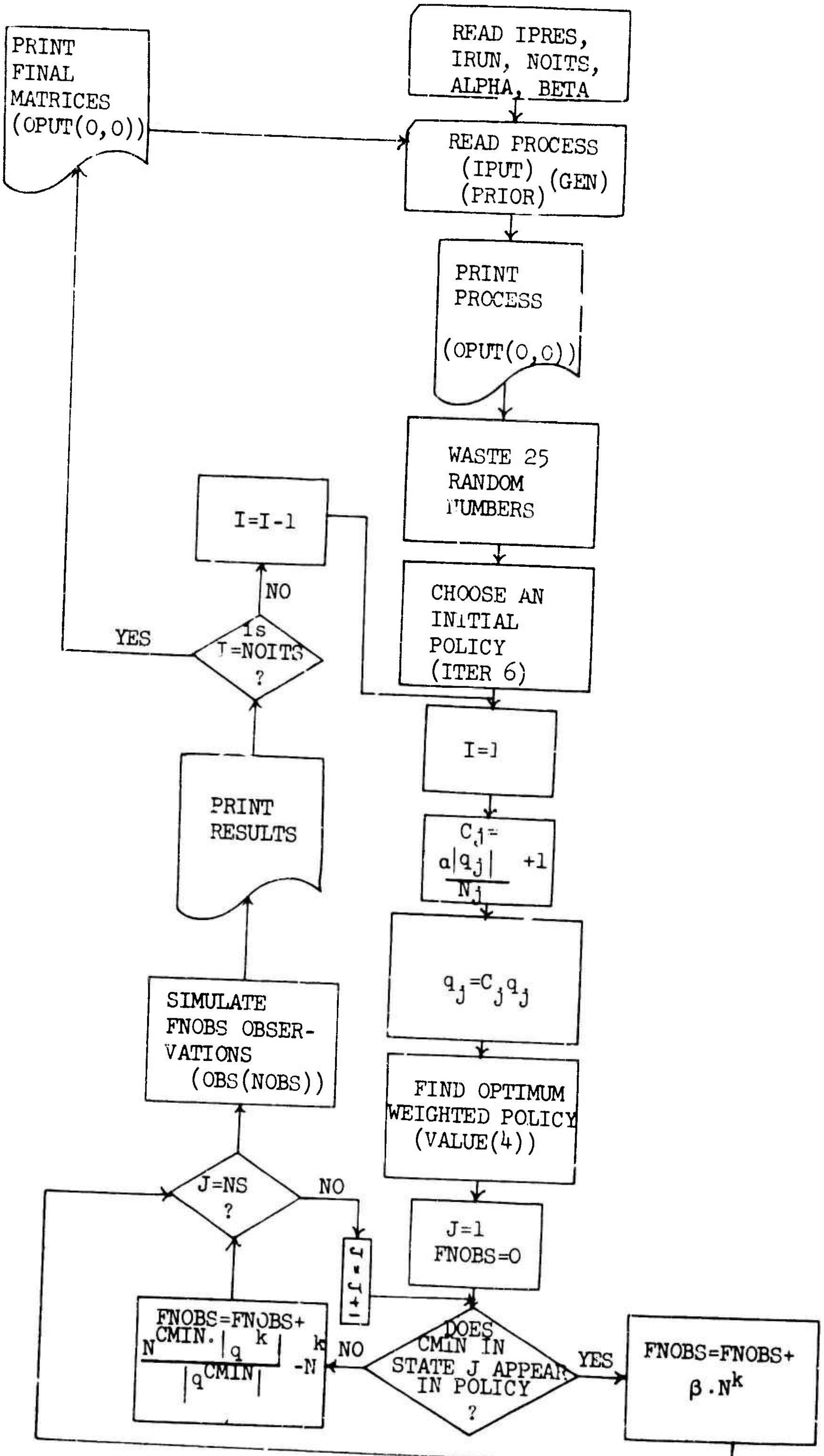
MAIN 1


READ PROCESS (IPUT)
READ IPRES, NOBS, NOITS, IRUN
PRINT PROCESS (OPUT(0,0))
Choose an initial policy to start calculations (IDUM=ITER(6))
WASTE 25 RANDOM NUMBERS
GENERATE A SAMPLE PROCESS (GEN)
I=1
FIND OPTIMAL POLICY OF EXPECTED PROCESS (GAIN=VALUE(4))
SIMULATE NOBS OBSERVATIONS UNDER THAT POLICY (OBS(NOBS))
PRINT I EFFICIENCY GAIN, POLICY (OPUT(I,N))
IS I=NOITS ?
YES
NO
I=I+1

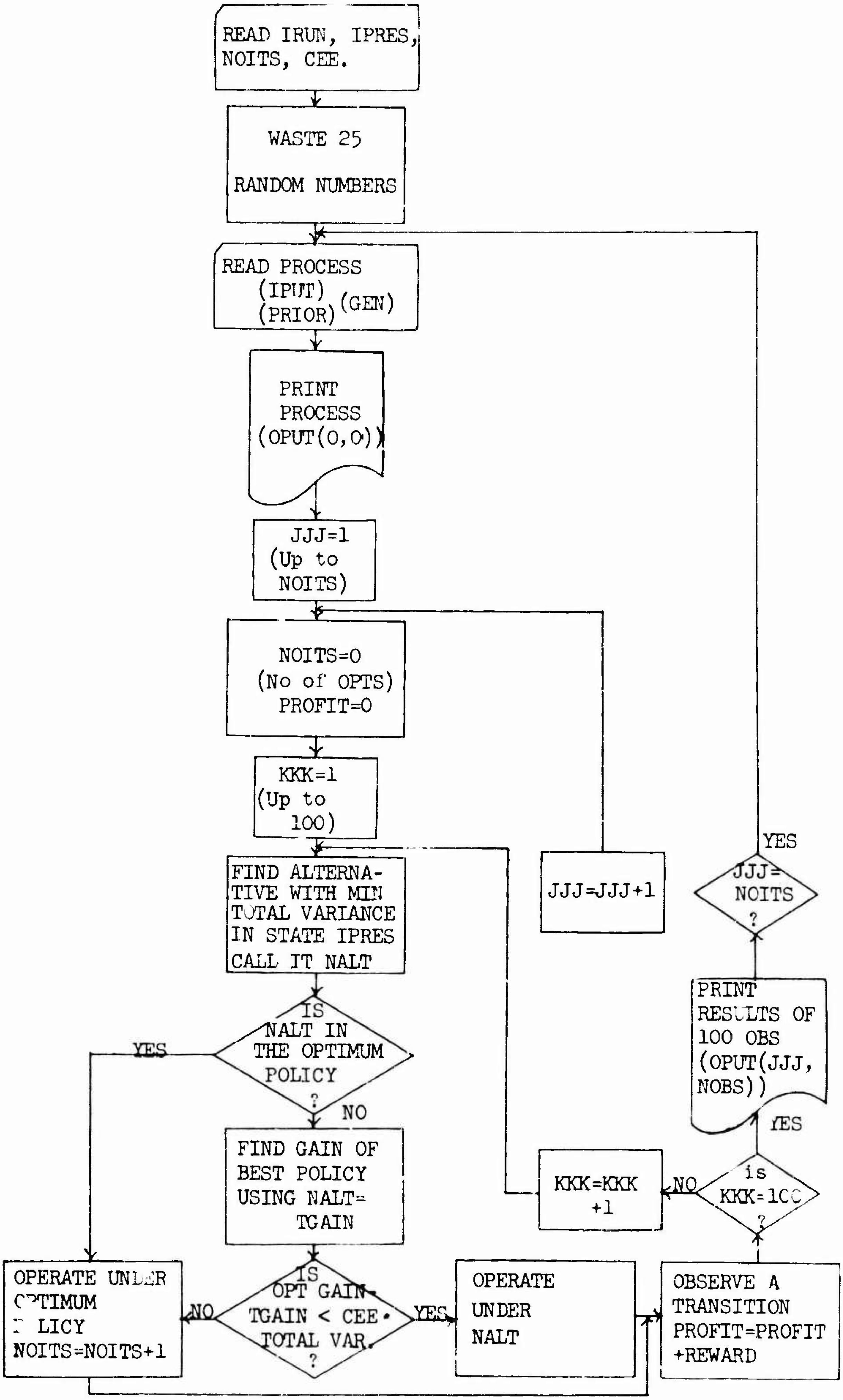
MAIN 2


WASTE 25 RANDOM NUMBERS
READ PROCESS (IPUT) (GEN) (PRIOR)
READ IPRES, IRUN, NOITS, ALPHA, BETA
PRINT PROCESS (OPUT(0,0))
KKK=1
FIND MINIMUM Q_i^k= QMIN
SAVE Q_i'S REPLACE BY $W_i=\frac{(Q_i-QMIN)^\alpha}{N_i}$
COMPUTE POLICY TO MAXIMIZE W_i IN EACH STATE(ITER(6))
FIND MINIMUM W_i= WMIN
RESTORE Q_i'S
NOBS= $\beta\cdot$ WMIN +1.
FIND GAIN OF THIS POLICY (GAIN=VALUE(3))
OBSERVE NOBS TRANSITIONS UNDER THIS POLICY (OBS(NOBS))
PRINT RESULTS
KKK= NOITS ?
YES
NO
KKK=KKK +1

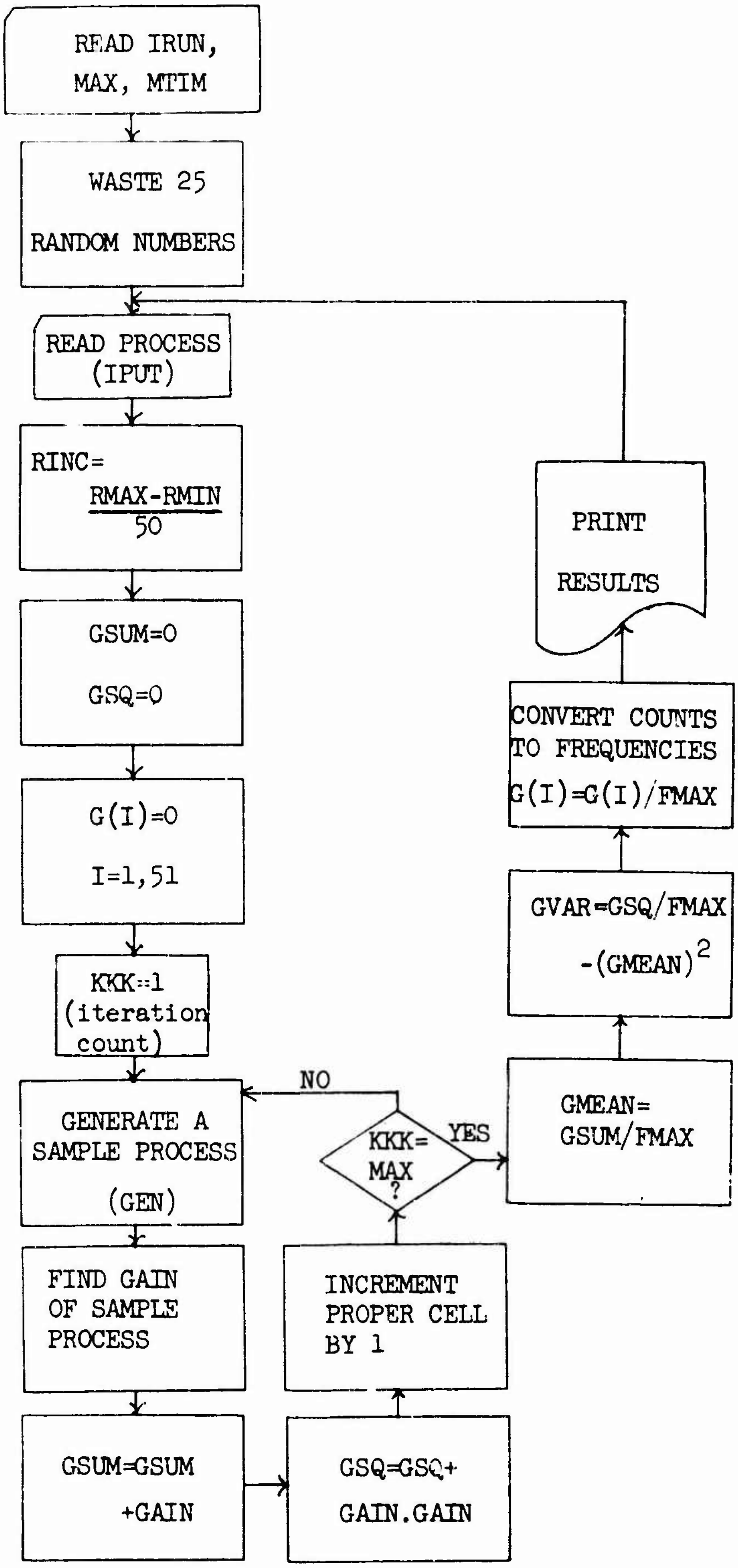
MAIN 3


READ IPRES, IRUN, NOITS, ALPHA, BETA
READ PROCESS (IPUT) (PRIOR) (GEN)
PRINT PROCESS (OPUT(0,0))
WASTE 25 RANDOM NUMBERS
CHOOSE AN INITIAL POLICY (ITER 6)
I=1
$C_j = \frac{a|q_j|}{N_j} + 1$
$q_j = C_j q_j$
FIND OPTIMUM WEIGHTED POLICY (VALUE(4))
J=1 FNOBS=0
DOES CMIN IN STATE J APPEAR IN POLICY ?
YES
FNOBS=FNOBS+ $\beta . N^k$
NO
FNOBS=FNOBS+ $N^{CMIN} . \frac{|q^k|}{|q^{CMIN}|} - N^k$
J=NS ?
NO
J = J+1
SIMULATE FNOBS OBSERVATIONS (OBS(NOBS))
PRINT RESULTS
Is I=NOITS ?
NO
I=I-1
YES
PRINT FINAL MATRICES (OPUT(0,0))

MAIN 4


READ IRUN, IPRES, NOITS, CEE.
WASTE 25 RANDOM NUMBERS
READ PROCESS (IPUT) (PRIOR) (GEN)
PRINT PROCESS (OPUT(0,0))
JJJ=1 (Up to NOITS)
NOITS=0 (No of OPTS) PROFIT=0
KKK=1 (Up to 100)
FIND ALTERNATIVE WITH MIN TOTAL VARIANCE IN STATE IPRES CALL IT NALT
IS NALT IN THE OPTIMUM POLICY ?
YES
NO
FIND GAIN OF BEST POLICY USING NALT= TGAIN
IS OPT GAIN-TGAIN < CEE• TOTAL VAR. ?
NO
YES
OPERATE UNDER OPTIMUM POLICY NOITS=NOITS+1
OPERATE UNDER NALT
OBSERVE A TRANSITION PROFIT=PROFIT +REWARD
is KKK=100 ?
NO
YES
KKK=KKK +1
PRINT RESULTS OF 100 OBS (OPUT(JJJ, NOBS))
JJJ= NOITS ?
YES
JJJ=JJJ+1

DISTRIBUTION OF GAIN



APPENDIX B

SUMMARY OF COMPUTER RUNS

B.1 Runs with MAIN I

B.1.1 Extreme point from sample process (Run I)

We chose as a prior the following matrix, $\{\bar{P}_{ij}^{k}\}$, with its corresponding set of prior parameters N_i^k:

		$\{\bar{P}_{ij}^{k}\}$			N_i^k
		$j=1$	$j=2$	$j=3$	
$i=1$	$k=1$	1/3	1/3	1/3	30
	$k=2$	1/10	8/10	1/10	20
	$k=3$	1/10	1/10	8/10	60
$i=2$	$k=1$	2/10	0	8/10	30
	$k=2$	8/10	1/10	1/10	20
$i=3$	$k=1$	1/3	1/3	1/3	30
	$k=2$	1/4	1/2	1/4	15
	$k=3$	8/10	1/10	1/10	40

As the actual process we used the taxicab example presented in the introduction. The actual process will be seen to be an "extreme point" from the prior. As expected (see 6.2), only policy (1, 1, 1) with an actual gain of 9.24 is ever chosen. The true optimum (2, 2, 2) with gain of 13.34, is never discovered.

B.1.2 Randomly chosen processes (Run 10)

This run was for a five-state process with three alternatives in each state. Both the prior probabilities and the rewards were drawn from random number tables. Each N_i was chosen to be 25, which we consider expresses a small degree of "certainty." The results are summarized in the table following. Notice that the optimal solution (or one very close to it) was always found quite rapidly, and that efficiency stays in the high 90's.

B.2 Runs with MAIN II

B.2.1 Run 2

The same data was used as in Run 1, but MAIN II was used to try to force (2, 2, 2) to be explored. Our parameters were $\alpha = 20$, $\beta = 100$. In 25 iterations, only policy (1, 1, 1) was chosen.

B.2.2 Run 3

We used the same data again, but modified our parameters to $\alpha = 2$, $\beta = 100$. Some experimentation was introduced: (1, 1, 1) was used for the first 19 iterations, then (1, 2, 1) with gain of 12.50 is tried for 6 iterations, then (1, 2, 2) with gain of 13.15 is tried once. But, then the process locks on (1, 1, 1).

50 ITERATIONS OF 20 OBSERVATIONS

RUN 10

MAIN 1

Sample Point	Optimal Policy	Optimal Gain	Max. Eff.	Iteration #	Min. Eff.	Iteration #	No. Opts.	Diff. policies tried
1	1, 3, 1, 2, 1	5.68	.9968	50	.9677	1	45	2
2	1, 3, 1, 2, 1	5.82	.9990	50	.9524	1	49	2
3	1, 1, 1, 2, 1	5.61	1.0000	1	.9886	2	38	3
4	1, 3, 1, 2, 3	5.83	.9873	50	.9783	1	0	2
5	1, 1, 1, 2, 1	6.20	1.000	50	1.000	1	50	1

B.3 Runs with MAIN III

B.3.1 F.un 4

Same data as Run 1, with parameter $\alpha = 20$.

Iteration	NOBS	Policy	Est. gain	Act. gain
1	54	221	8.69	8.81
2	111	111	9.38	9.20
3	34	222	5.70	13.34
4	18	222	9.26	13.34
5	8	222	10.57	13.34
6	8	222	11.76	13.34
7	5	222	10.90	13.34
8	5	222	11.12	13.34
9	4	222	11.14	13.34
10	4	222	11.29	13.34
11	40	212	8.95	8.81

Locks on 222

B.3.2 Run 5: MAIN III, same data as Run 1, $\beta = 0.4$, $\alpha = 20$

Iteration	NOBS	Policy	Est gain	Act gain
1	54	212	8.69	8.81
2	133	111	9.38	9.20
3	40	222	5.70	13.34
4	19	222	9.46	13.34
5	37	111	10.33	13.34
6	62	212	8.99	8.81
7	85	222	11.07	-
8	138	212	9.09	-
9	287	122	12.05	13.15
10	863	111	9.29	-
11	233	222	12.85	-
12	287	222	13.04	-
13	708	212	8.99	-
14	620	222	13.24	-
15	1418	122	13.04	13.15
16	2004	323	8.93	8.98
17	4406	111	9.24	-

18 Locks on 222 (no. of observations at each iteration grows larger and larger)

B.3.3 Run 6

Same data as Run 5, but parameters $\alpha = 20$, $\beta = 0.1$. This now uses (2, 2, 2), most of the time, but converges slowly to it. Begins with (2, 1, 2), then (1, 1, 1), and later returns to (2, 1, 2) for a few iterations. Convergence is slow.

B.3.4 Run 7

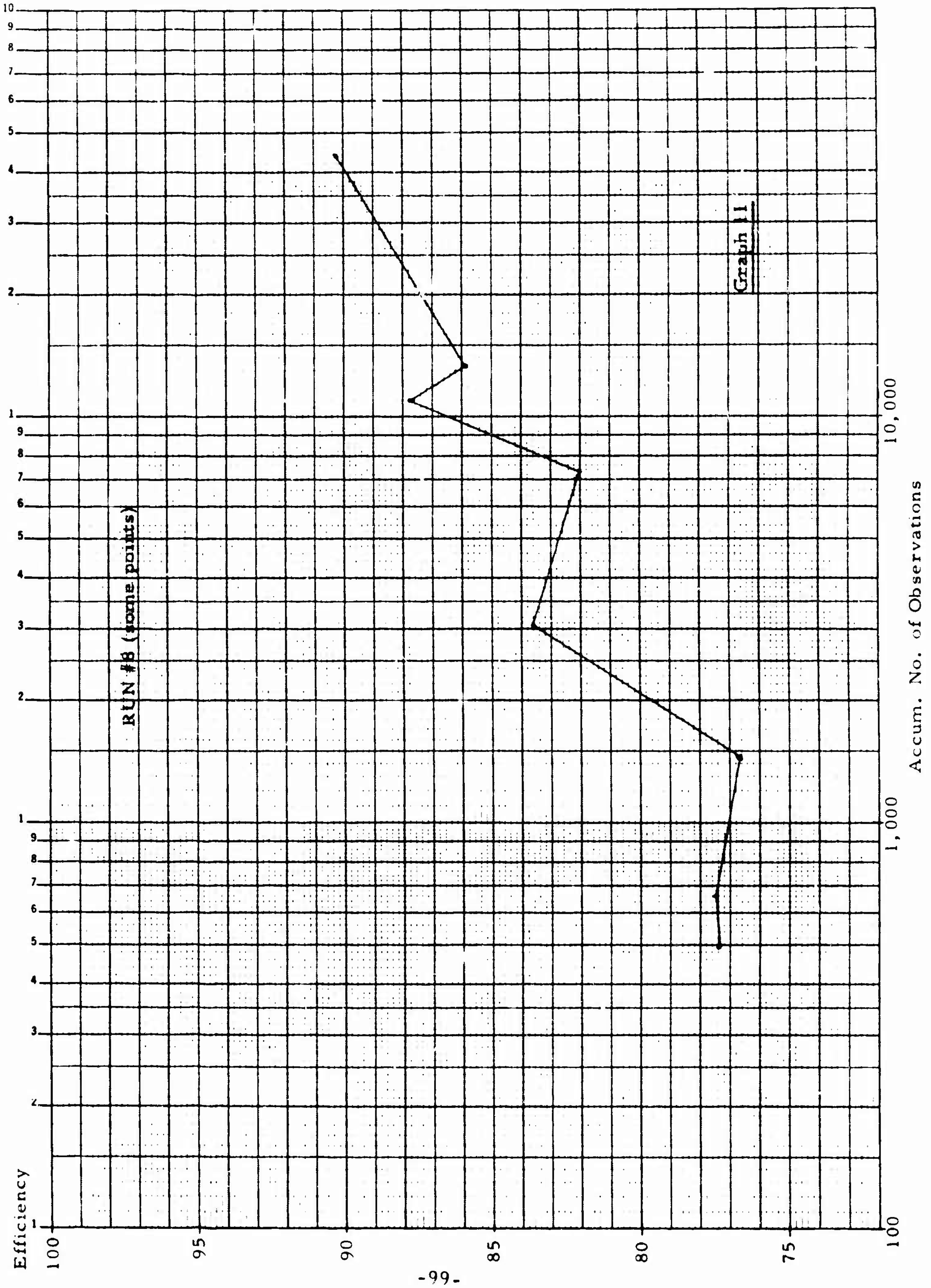
Same data as Run 6, with $\alpha = 20$, $\beta = .01$. Same general shape as Run 6, but there are more observations in later stages.

B.3.5 Run 8

More weight is given to immediate returns by making $\alpha = 40$, $\beta = .1$. See graph following for plot of efficiency (see Chapter VI) versus accumulated number of observations.

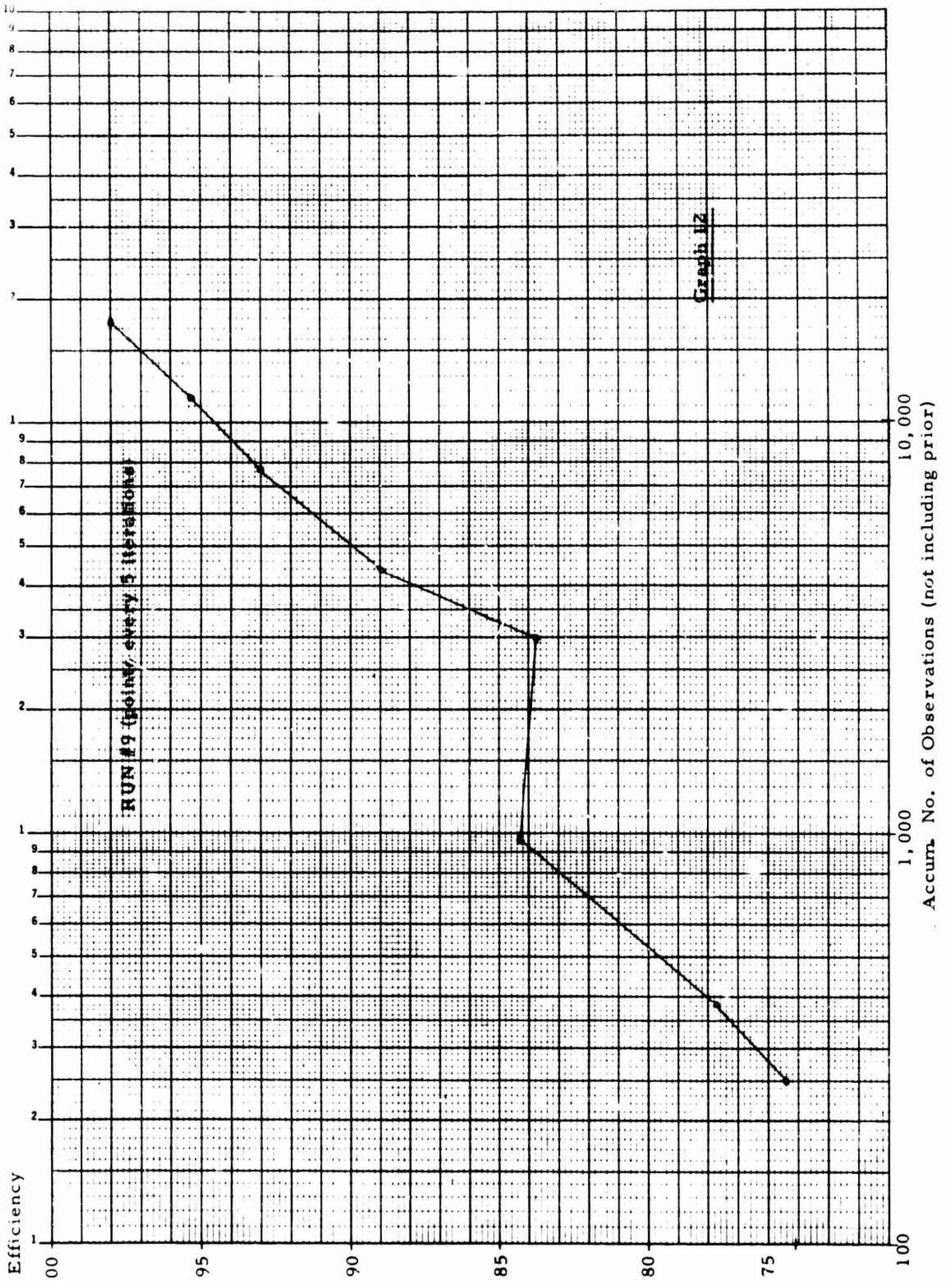
B.3.6 Run 9

Same data. $\alpha = 10$, $\beta = 1$. See graph following.



Graph 11

RUN #8 (some points)



Graph 17

RUN #9 (point every 5 iterations)
Efficiency
100
95
90
85
80
75
Accum. No. of Observations (not including prior)
100
1,000
10,000
1
2
3
4
5
6
7
8
9
10

APPENDIX C

FINDING A SECOND BEST POLICY

C. 1 Motivation

In Chapter VII we discussed the advantages of being able to select the ten best policies for the expected process, in order to simulate and find more explicit information about these most-likely candidates. But finding the ten best policies hinges upon finding an algorithm to find the second best policy.

C. 2 Impossibility of exact solution

Let O denote the optimal policy of a known process (such as the expected process), and let N denote the next, or second, best policy. Then :

$$g^{\Delta} = g^{O} - g^{N}$$

is to be a minimum for $N \neq O$.

Define:

$$\gamma_i^{\Delta} = [q_i^O + \sum_j p_{ij}^O v_j^O] - [q_i^N + \sum_j p_{ij}^N v_j^O]$$

We know that:

$$g^O + v_i^O = q_i^O + \sum_j p_{ij}^O v_j^O$$

$$g^N + v_i^N = q_i^N + \sum_j p_{ij}^N v_j^N$$

Thus:

$$g^O - g^N + v_i^O - v_i^N = \gamma_i^N + \sum_j p_{ij}^N v_i^O - \sum_j p_{ij}^N v_i^N$$

Multiplying both sides by π_i^N and summing over i, we have:

$$\sum_i (g^O - g^N) \pi_i^N + \sum_i v_i^O \pi_i^N - \sum_i v_i^N \pi_i^N = \sum_i \gamma_i^N \pi_i^N$$

$$- \sum_i \sum_j p_{ij}^N \pi_i^N v_j^O - \sum_i \sum_j p_{ij}^N \pi_i^N v_j^N$$

But we recall that $\pi p = \pi$, whence,

$$\sum_i \sum_j p_{ij}^N \pi_i^N v_j^O = \sum_j v_j^O \pi_j^N$$

$$\sum_i \sum_j p_{ij}^N \pi_i^N v_j^N = \sum_j v_j^N \pi_j^N$$

So finally,

$$g^O - g^N = g^\Delta = \sum_i \gamma_i^N \pi_i^N$$

Thus, the second best policy is that which minimizes:

$$\sum_i \pi_i^N \gamma_i^N \qquad N \neq O$$

But minimizing this quantity involves a search through all possible sets of π_i^N and this is prohibitive.

C. 3 Approximations to second best policy

We can get an "approximation" by changing only one state from the optimal policy, so that

$$\gamma_i^N = 0 \quad i \neq k$$
$$\gamma_k^N = \text{min possible for that state} > 0$$

This procedure serves to minimize:

$$\sum_i \gamma_i^N$$

but ignores the weighting by the π 's.

C. 4 Justification of approximation

This approximation procedure may be used as follows:

1. Find optimum policy.

2. Perturb each state: i.e., find N policies, each differing from the optimal in one state. Make best such perturbation possible in each state (that is minimize γ_i).

3. Apply mean-variance algorithm to these N policies.

The obvious disadvantage of this approach is that the N policies so determined are not actually the N best. But a compensating advantage is that at least 2 alternatives in each state are always considered, which encourages wide consideration of experimentation. Also, since the N best policies of the expected process are not necessarily the N best of the actual process, it is not so critical to find the precise N best of the expected process, so long as the ones chosen embrace a wide variety of alternatives, and are reasonable candidates for inspection. This approximation technique meets these demands.

BIBLIOGRAPHY

1. Howard, R. A., *Dynamic Programming and Markov Processes,* Wiley, New York, 1960

2. Mosimann, J. E., "On the Compound Multinominal Distribution," *Biometrika,* Vol. 49 (1962), pp. 65-82

3. Silver, E. A., "Markovian Decision Processes with Uncertain Transition Probabilities or Rewards," *Technical Report No. 1,* Operations Research Center, M.I.T., 1963

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1 ORIGINATING ACTIVITY (Corporate author)

Operations Research Center
Massachusetts Institute of Technology
Cambridge, Mass. 02139

2a REPORT SECURITY CLASSIFICATION

Unclassified

2b GROUP

3 REPORT TITLE

Technical Report No. 11
"MARKOVIAN DECISION PROCESSES WITH UNKNOWN TRANSITION PROBABILITIES"

4 DESCRIPTIVE NOTES (Type of report and inclusive dates)

Technical Report

5 AUTHOR(S) (Last name, first name, initial)

Cozzolino, John M.
Miller, Ralph L.
Gonzalez-Zubieta, Romulo

6 REPORT DATE March 1965	7a TOTAL NO. OF PAGES 105	7b NO. OF REFS 3

8a. CONTRACT OR GRANT NO

Nonr-1841(87)

b. PROJECT NO

NR 042-230

c.

d.

9a. ORIGINATOR'S REPORT NUMBER(S)

(Technical Report No. 11)

9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)

10 AVAILABILITY/LIMITATION NOTICES

11. SUPPLEMENTARY NOTES

12. SPONSORING MILITARY ACTIVITY

Office of Naval Research, Branch of Logistics and Mathematical Statistics

13. ABSTRACT

A dynamic programming formulation for the Markovian decision process when transition probabilities are unknown is proposed. This formulation is used to solve simple problems, but is shown to be too difficult to apply to more complex systems.

Various approximate methods are then proposed and discussed. A simple approximating algorithm is finally presented.

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Markov						
decision						
Bayesian						
transition						
beta distribution						
prior						
dynamic programming						
programming						
knowledge space						
simulation						
policy						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization (*corporate author*) issuing the report.

2a. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8a. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report

9b. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers (*either by the originator or by the sponsor*), also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through
_______________________________."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through
_______________________________."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through
_______________________________."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring (*paying for*) the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS)*, *(S)*, *(C)*, *or (U)*.

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, rules, and weights is optional.